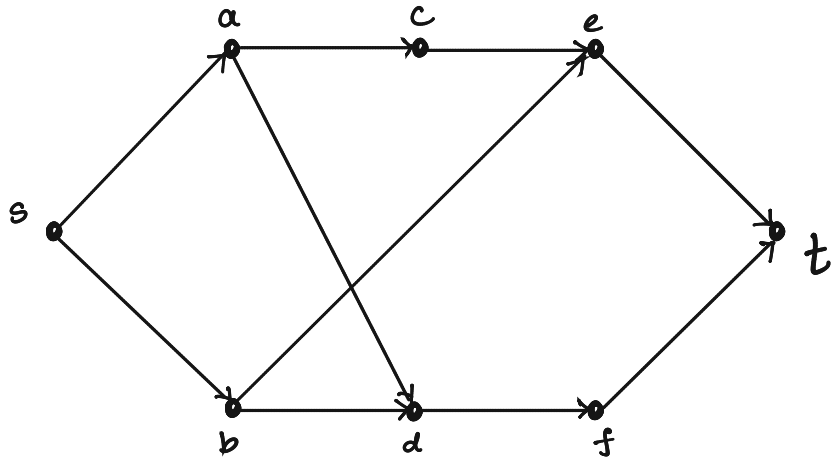
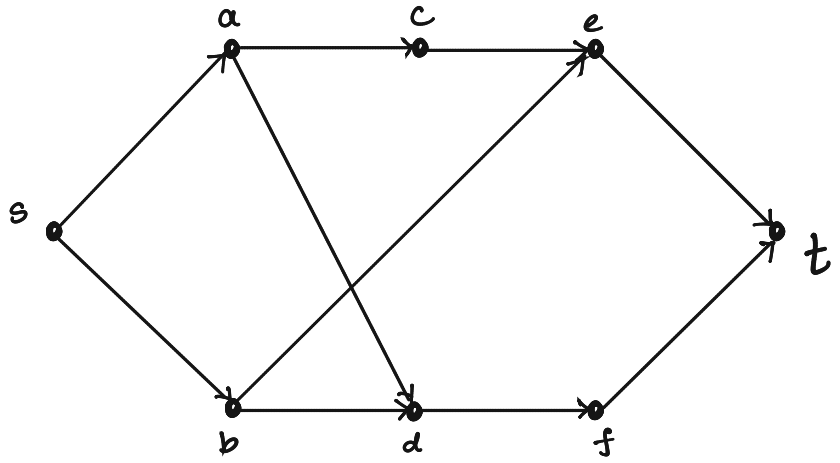


A DIRECTED GRAPH

s ← SOURCE

t ← SINK

$\{a, b, c, d, e, f\}$ ← INTERNAL NODES

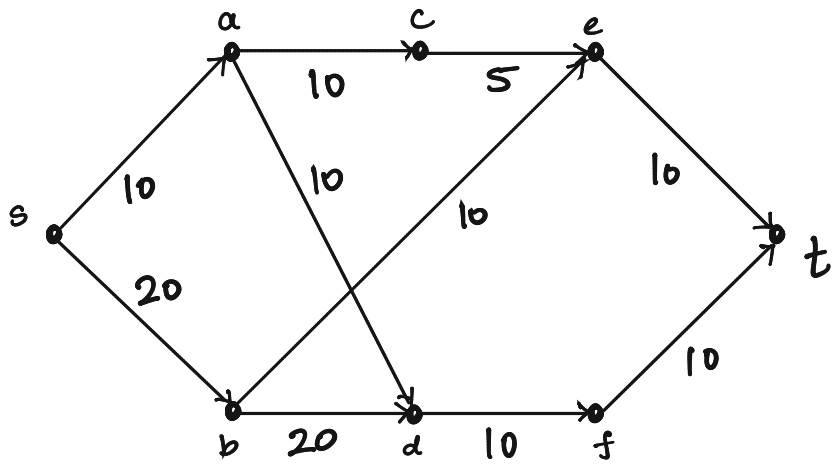


A DIRECTED GRAPH

s ← SOURCE [NO INCOMING EDGE]

t ← SINK [NO OUTGOING EDGE]

{a, b, c, d, e, f} ← INTERNAL NODES



A DIRECTED GRAPH

s ← SOURCE [NO INCOMING EDGE]

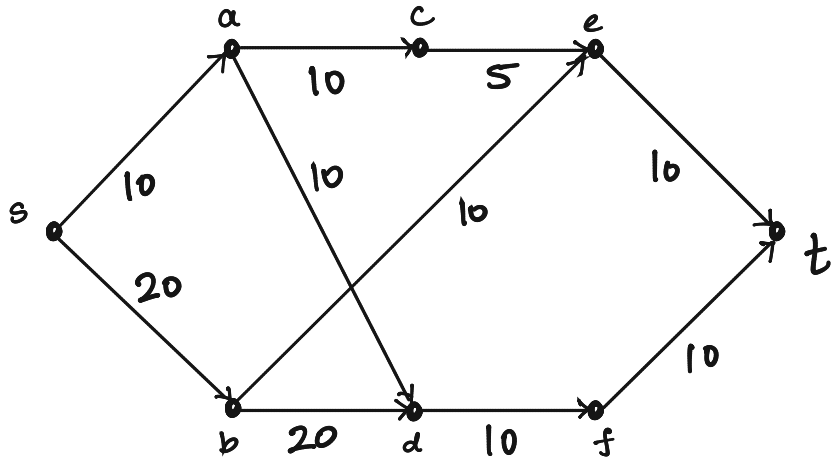
t ← SINK [NO OUTGOING EDGE]

$\{a, b, c, d, e, f\}$ ← INTERNAL NODES

EACH EDGE HAS SOME CAPACITY

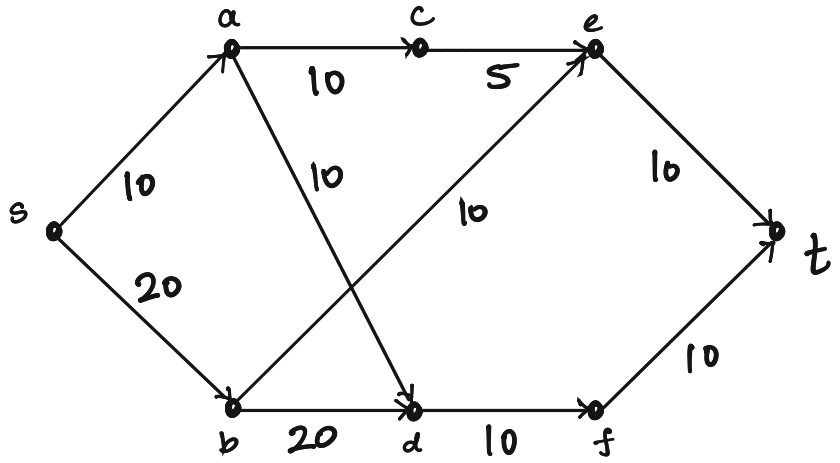
$c : E \to \mathbb{R}^+$

$c(e)$ ← CAPACITY OF EDGE e



→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY.

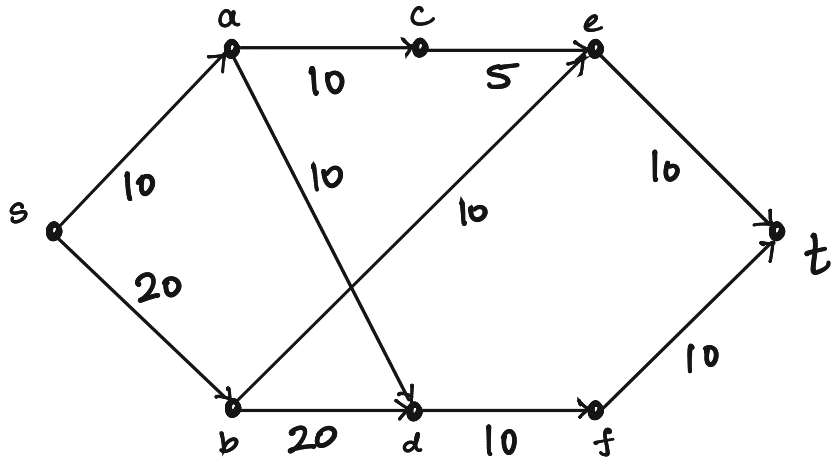


→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY

OBJECTS SENT THROUGH e

$\leq c(e)$



→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY

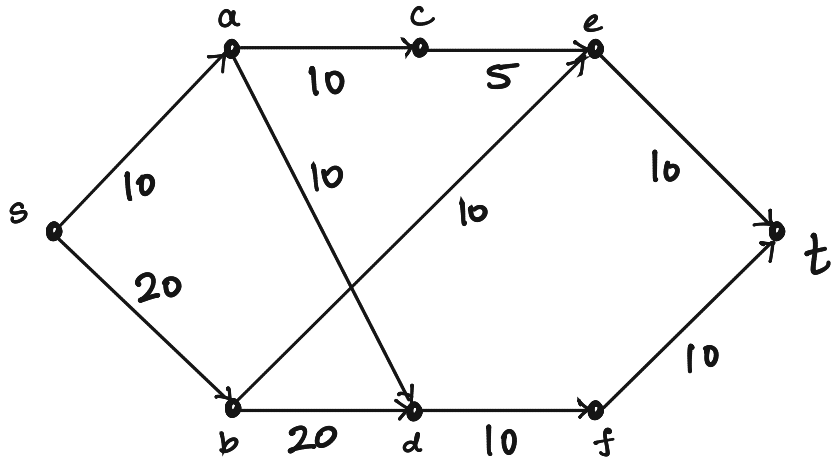
OBJECTS SENT THROUGH e

$\leq$ c(e)

f(e) $\leq$ c(e) (CAPACITY CONSTRAINT)

↑

FLOW THROUGH e.



→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY

OBJECTS SENT THROUGH e

$\leq$ c(e)

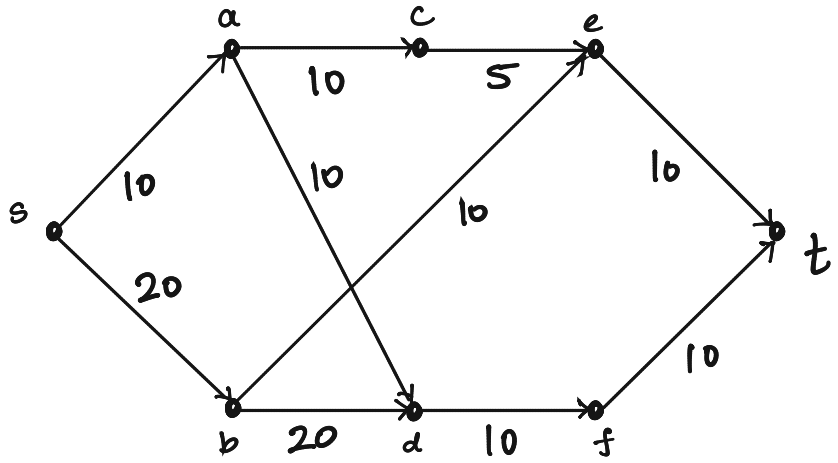
f(e) $\leq$ c(e) (CAPACITY CONSTRAINT)

↑

FLOW THROUGH e.

→ # OBJECTS RECIVED BY v

= # OBJECTS SENT OUT BY v.



→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY

OBJECTS SENT THROUGH e

$\leq$ c(e)

f(e) $\leq$ c(e) (CAPACITY CONSTRAINT)

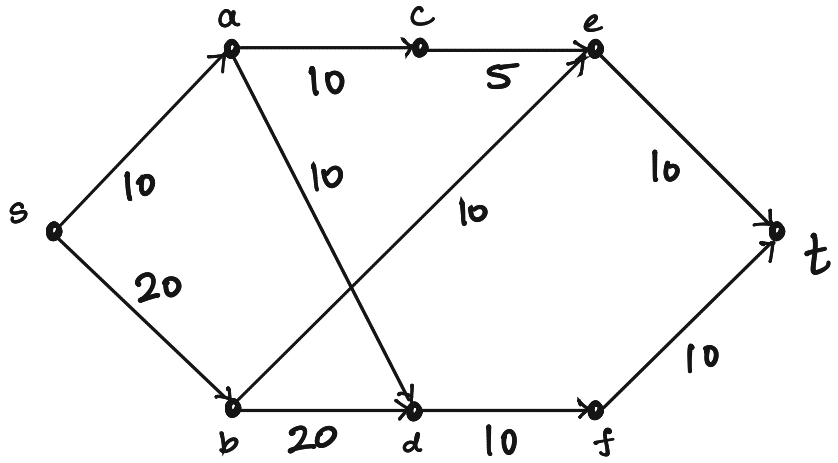
↑

FLOW THROUGH e.

→ # OBJECTS RECIVED BY v

= # OBJECTS SENT OUT BY v.

(TRUE FOR ALL VERTICES EXCEPT s & t)



→ s WANTS TO SEND SOME OBJECTS TO t.

→ # OBJECT SENT ON AN EDGE CANNOT EXCEED ITS CAPACITY

OBJECTS SENT THROUGH e

$\leq$ c(e)

$f(e) \leq c(e)$ (CAPACITY CONSTRAINT)

↑

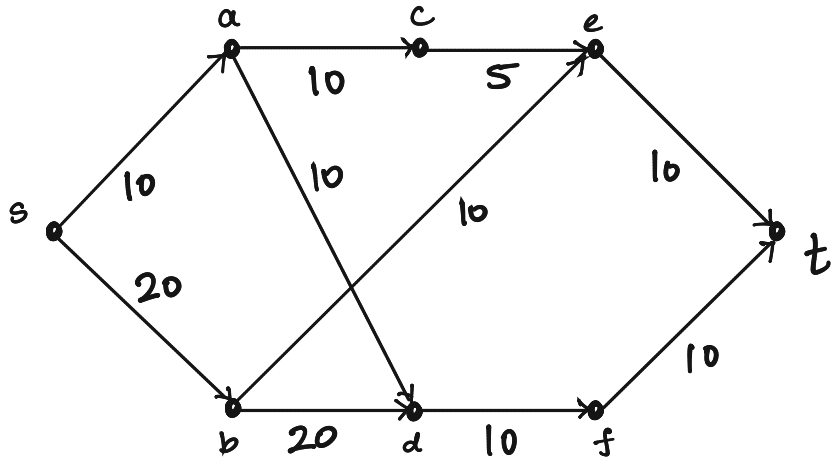
FLOW THROUGH e.

→ # OBJECTS RECIVED BY v

= # OBJECTS SENT OUT BY v.

(TRUE FOR ALL VERTICES EXCEPT s & t)

$$\sum_{e \text{ INCOMING TO } v} f(e) = \sum_{e \text{ OUTGOING FROM } v} f(e)$$ (CONSERVATION OF FLOW)



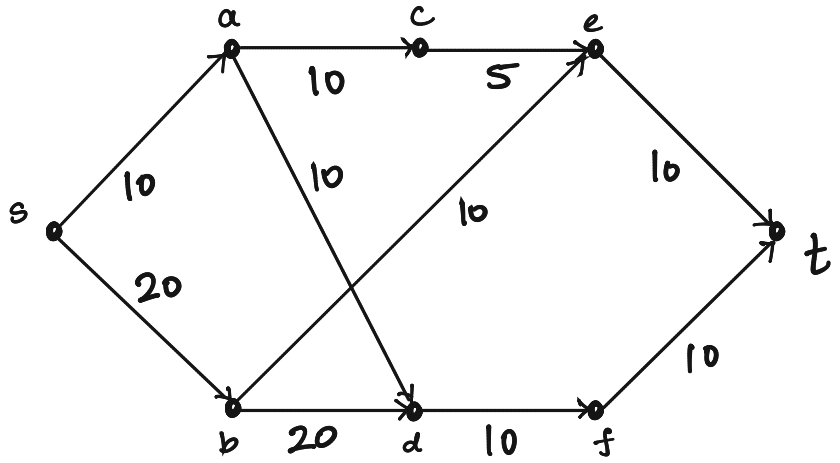
FIND THE MAXIMUM FLOW FROM s to t

SUBJECT TO

(1) $f(e) \leq c(e)$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ INCOMING TO } v} f(e) = \sum_{e \text{ OUTGOING FROM } v} f(e)$

(CONSERVATION OF FLOW)

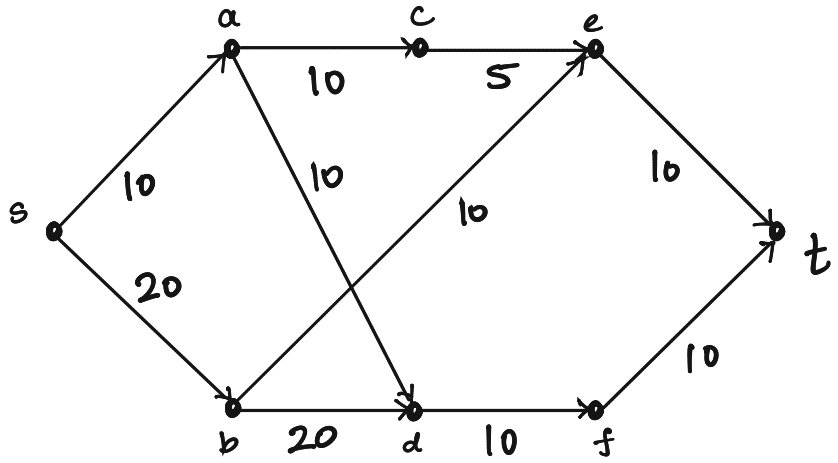


maximize $\sum_{e \text{ OUTGOING FROM } s} f(e)$

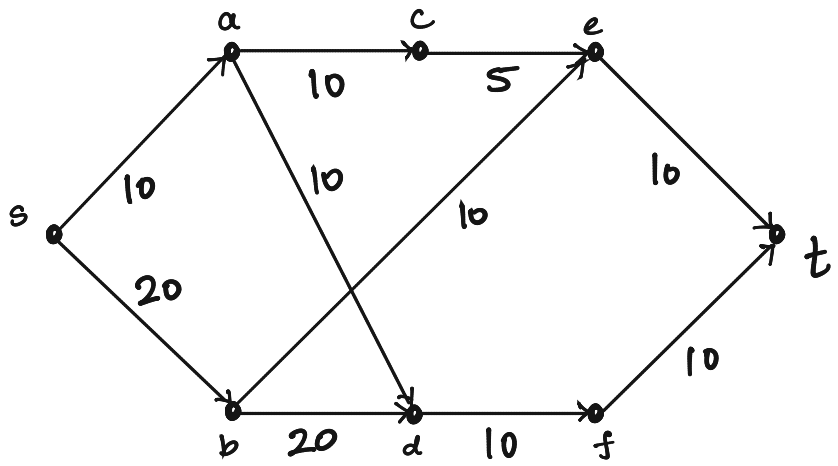
(1) $f(e) \leq c(e)$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ INCOMING TO } v} f(e) = \sum_{e \text{ OUTGOING FROM } v} f(e)$

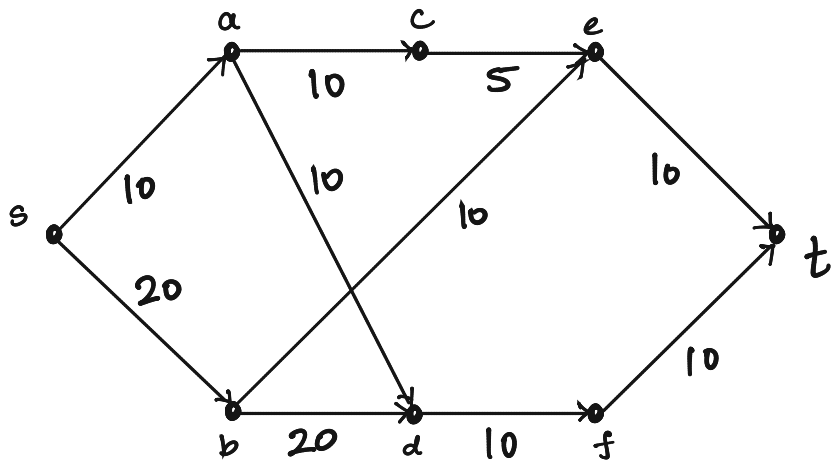
(CONSERVATION OF FLOW)



BE GREEDY :

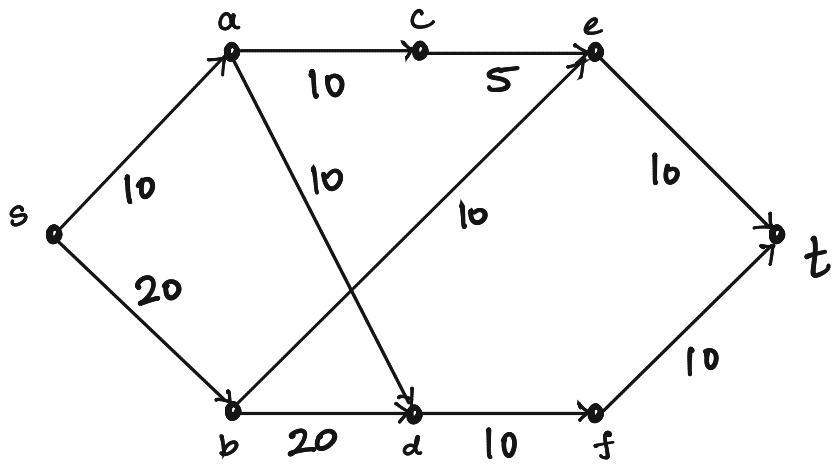


BE GREEDY: "PUSH" AS MUCH FLOW AS YOU CAN ALONG A PATH



BE GREEDY: "PUSH" AS MUCH FLOW AS YOU CAN ALONG A PATH

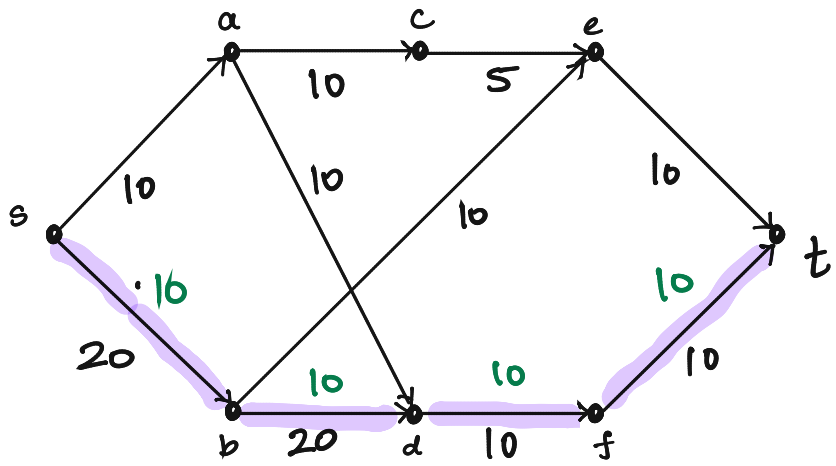
→ FIND A st PATH THAT MAXIMIZES THE MINIMUM EDGE WEIGHT ON THE PATH

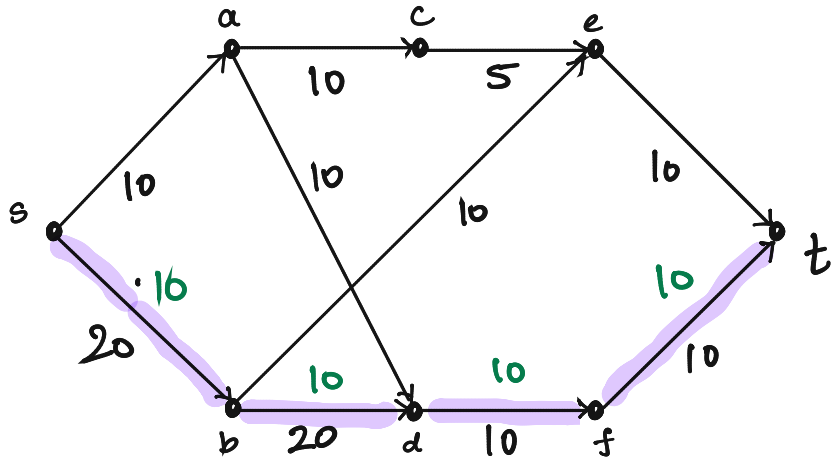


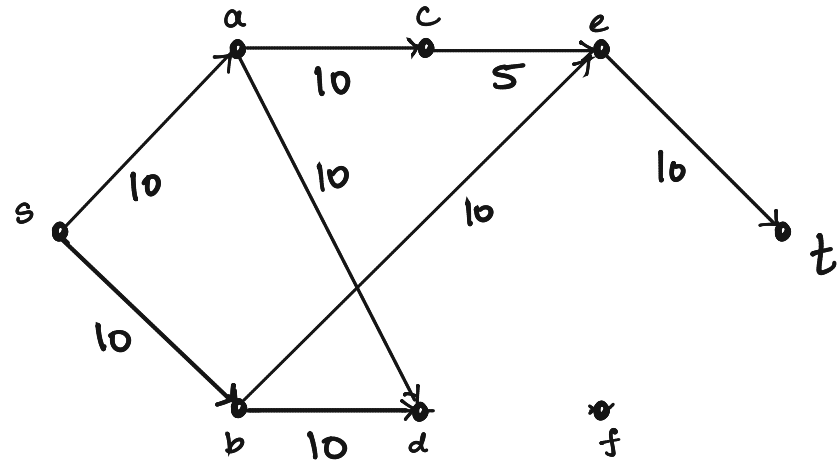
BE GREEDY: "PUSH" AS MUCH FLOW AS YOU CAN ALONG A PATH

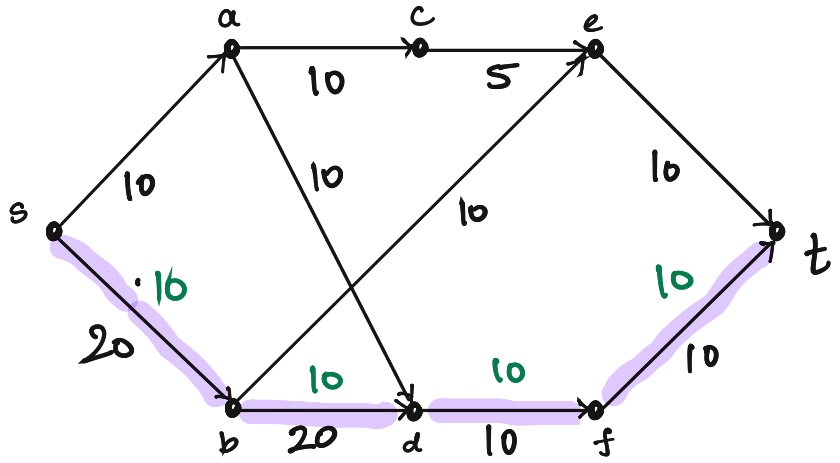
→ FIND A st PATH THAT MAXIMIZES THE MINIMUM EDGE WEIGHT ON THE PATH

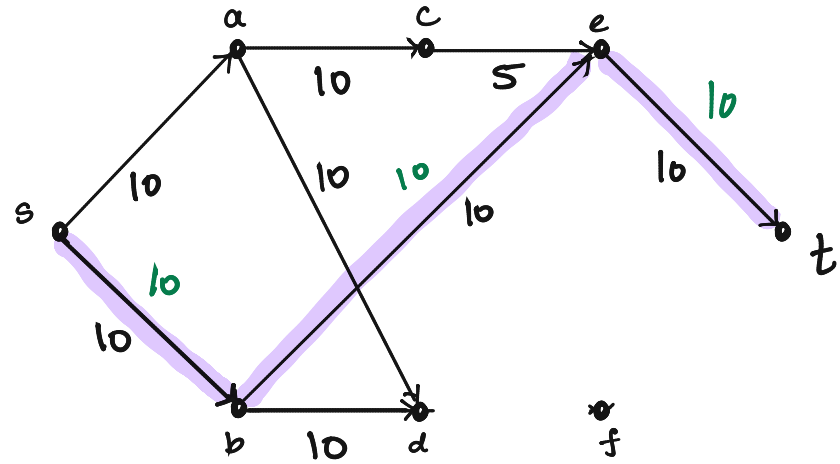
→ MAXIMUM BOTTLENECK PATH FROM s to t.

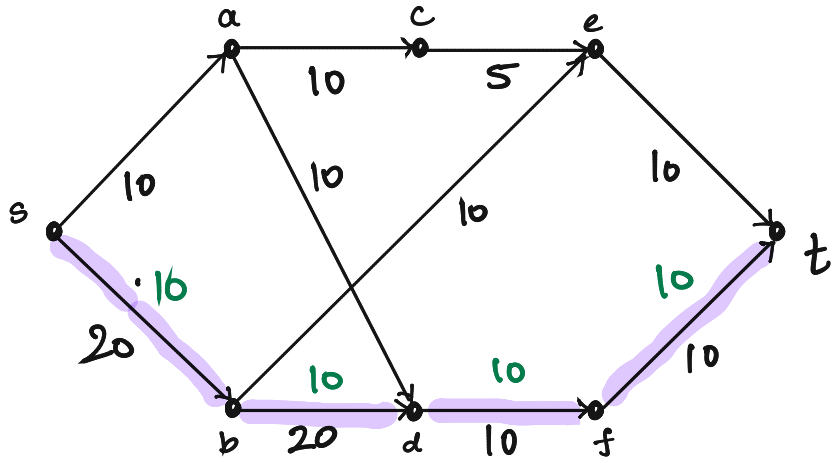

a
c
e
10
5
10
10
10
10
s
t
16
10
20
10
10
10
b
20
d
10
f

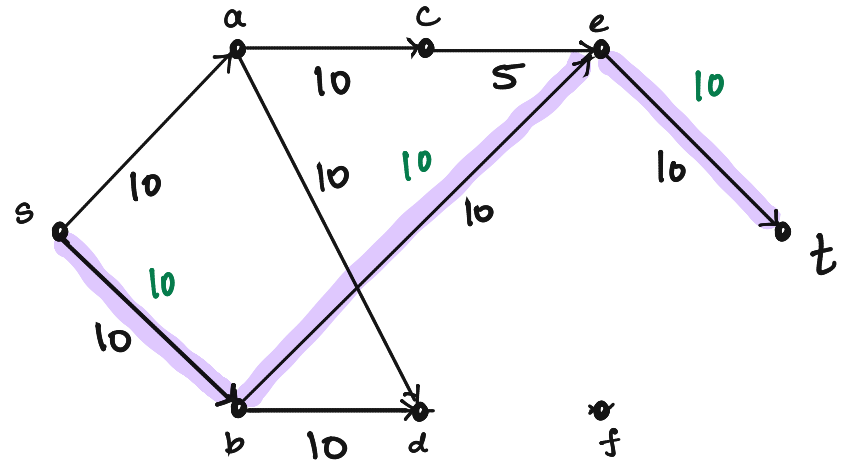

a
c
e
10
5
10
10
10
s
10
t
16
10
20
10
10
10
b
20
d
10
f

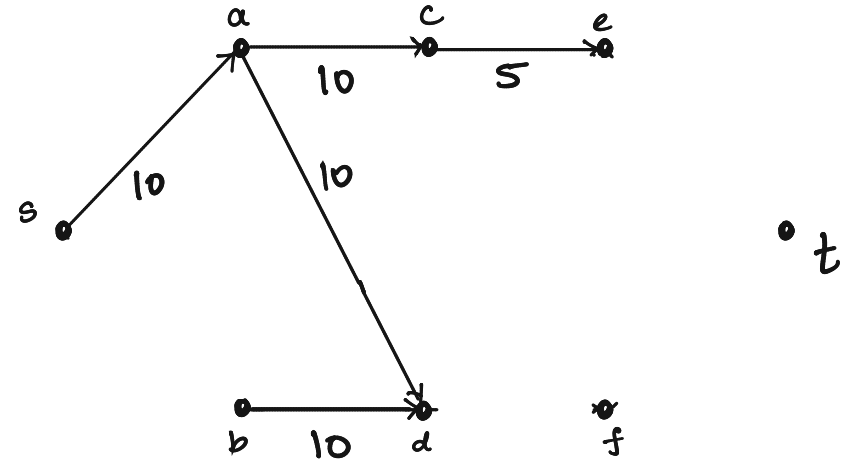

a
c
e
10
5
10
10
10
s
10
t
10
b
10
d
f

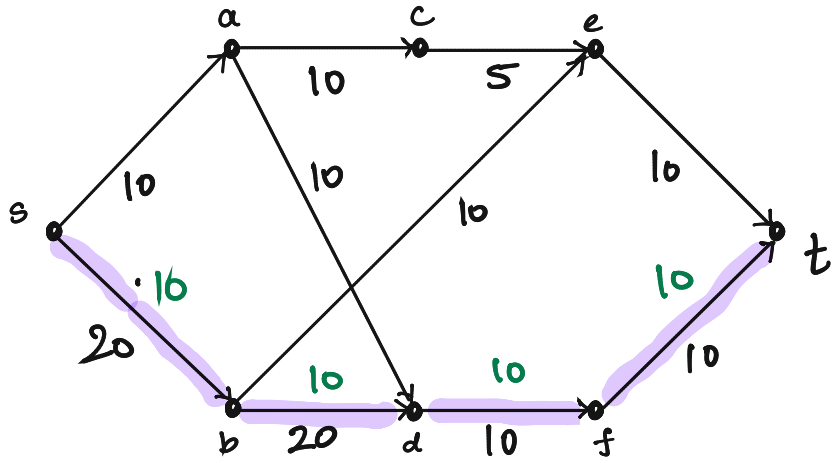

a
c
e
10
5
10
10
10
s
10
t
10
10
20
10
10
10
b
20
d
10
f

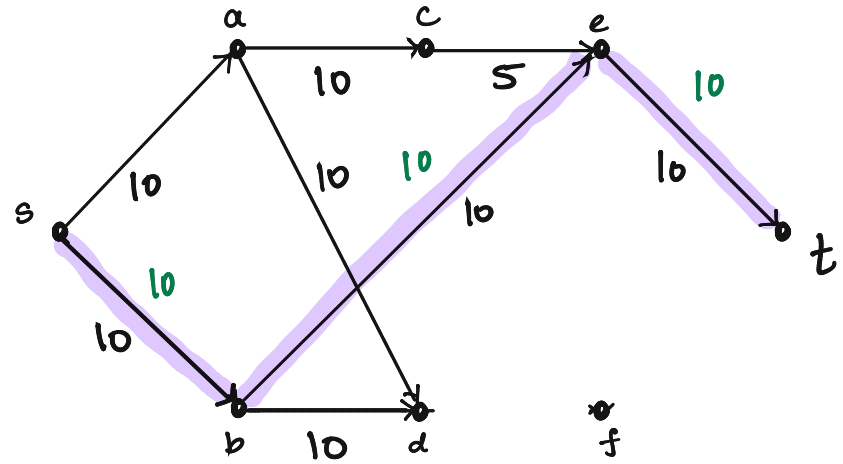

a
c
e
10
5
10
10
10
10
s
10
10
t
10
10
b
10
d
f

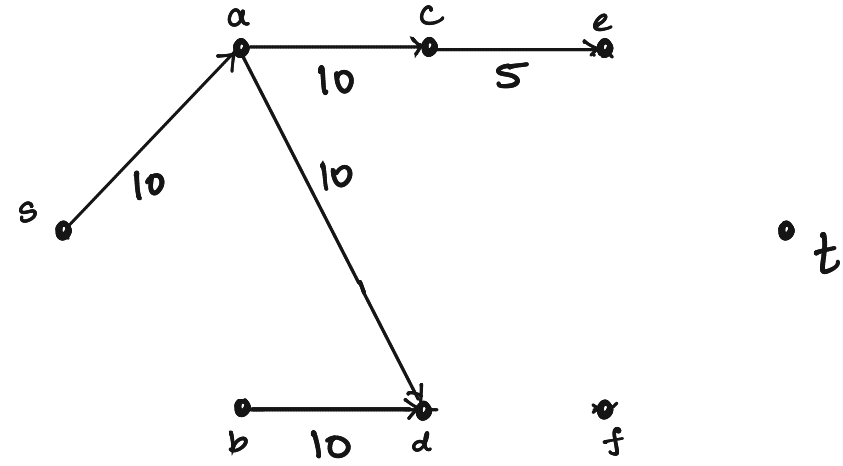

a
c
e
10
5
10
10
10
s
10
t
16
10
20
10
10
10
b
20
d
10
f


a
c
e
10
5
10
10
10
10
s
10
t
10
10
b
10
d
f


a
c
e
10
5
10
10
s
t
b
10
d
f

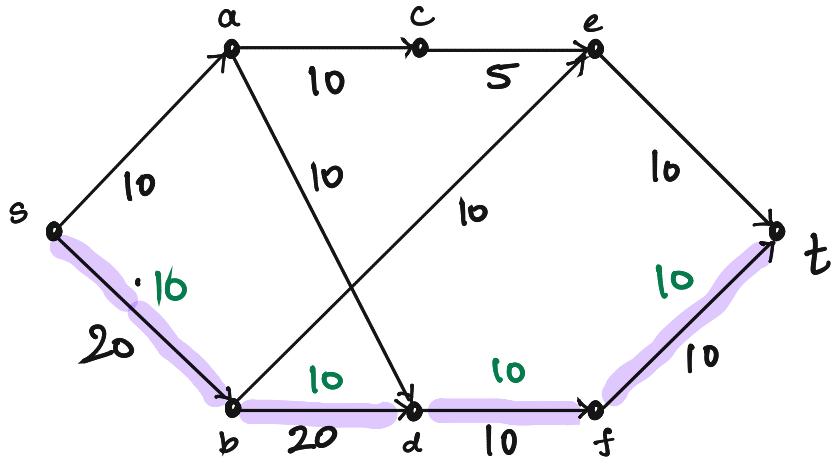


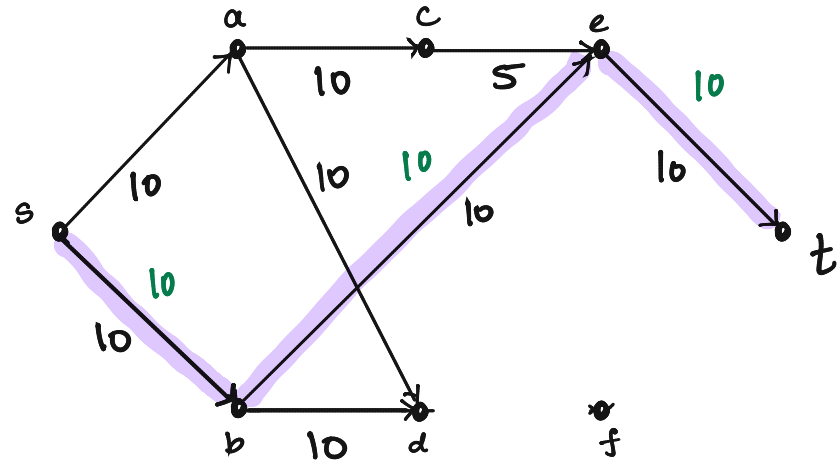


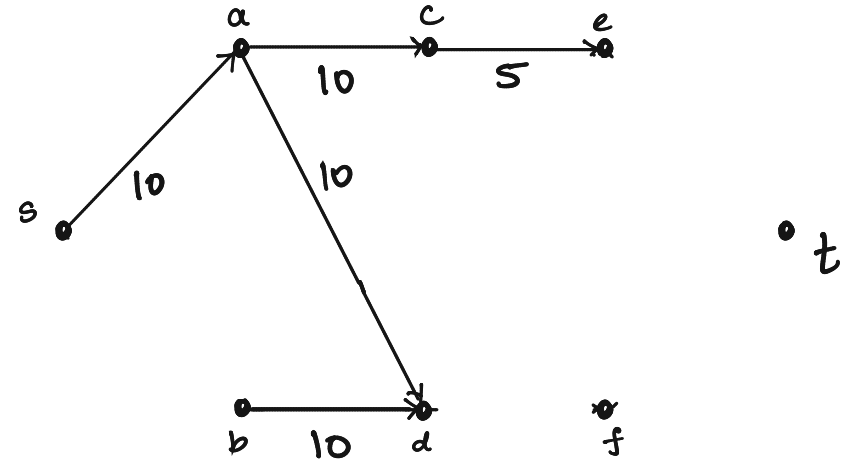


→ NO PATH LEFT FROM s to t

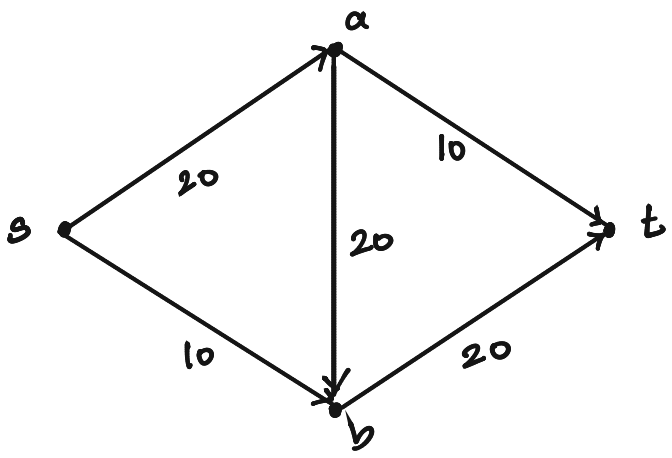
→ PUSHED A FLOW OF 20.

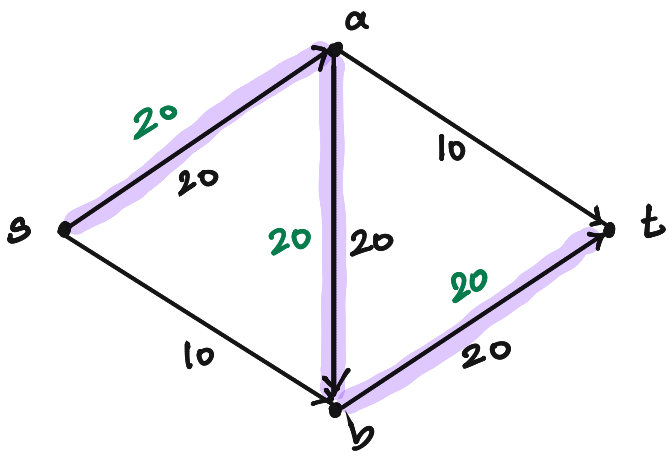


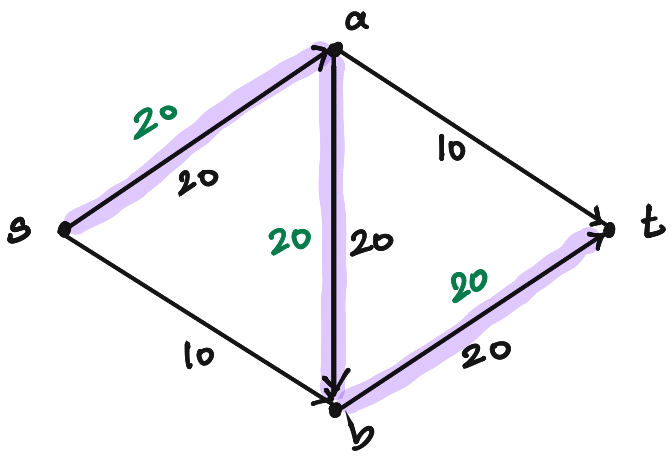


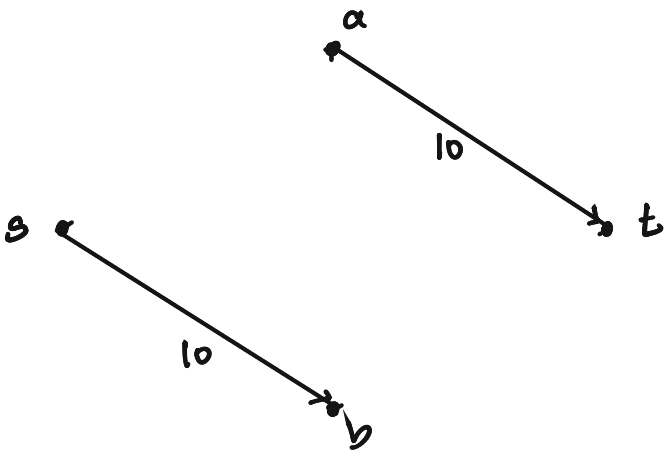


Q: Is this algorithm correct?

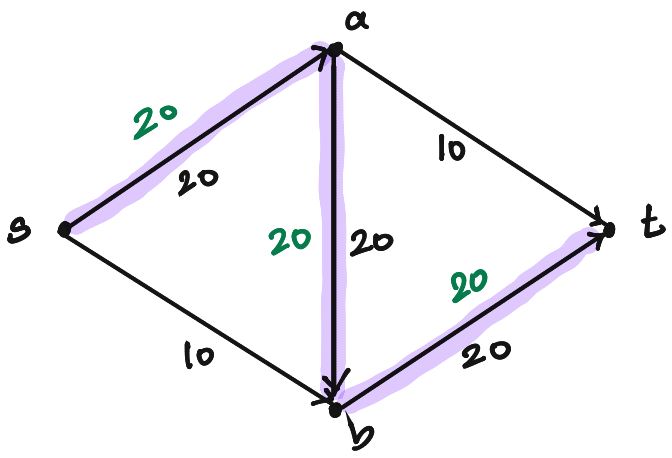

a
s
t
b
20
10
20
10
20

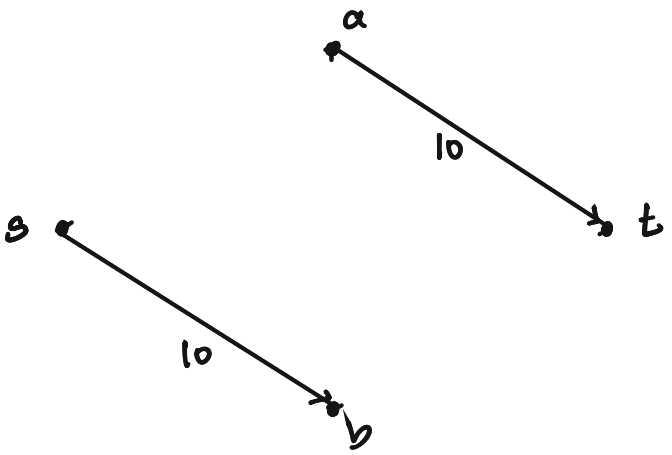

a
s
t
b
20
20
10
20
20
20
10
20





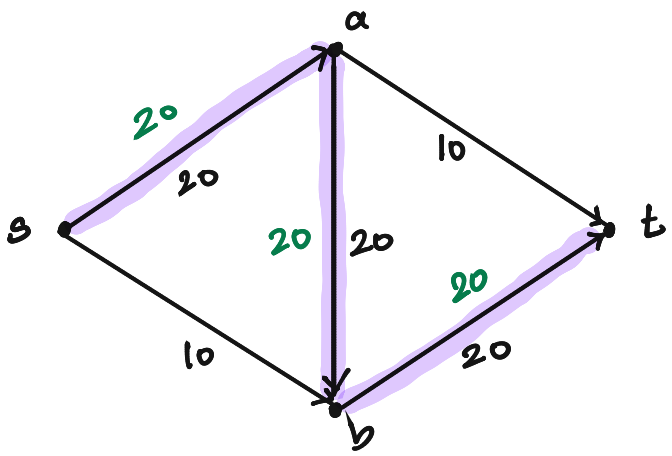
NO MORE PATH FROM s to t.

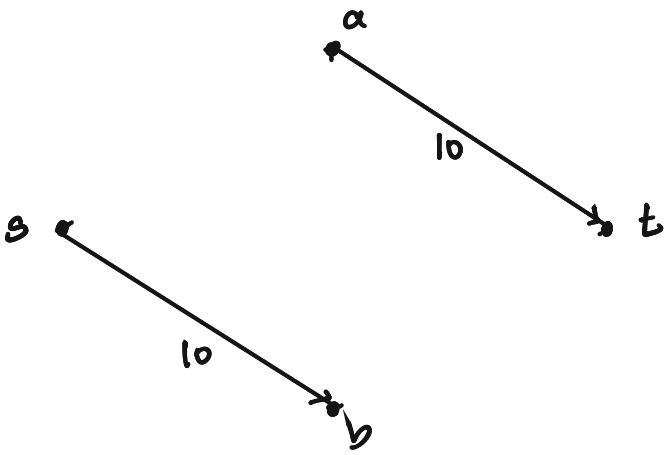




NO MORE PATH FROM s to t.

FLOW OUT OF S : 20.

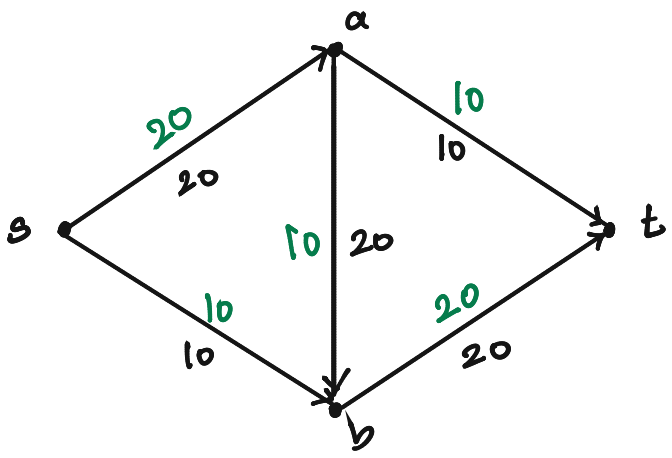




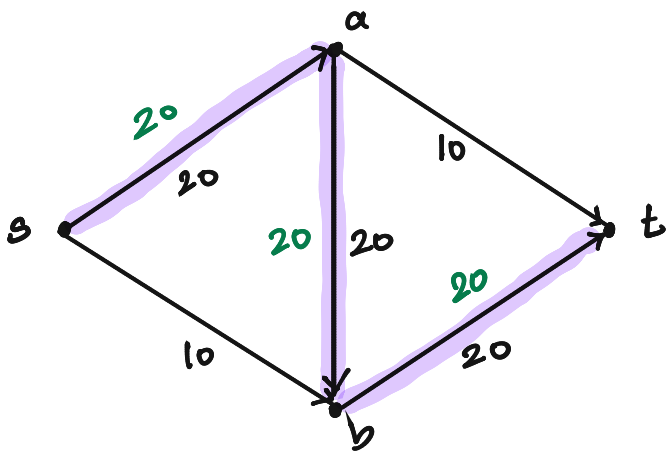
NO MORE PATH FROM s to t.

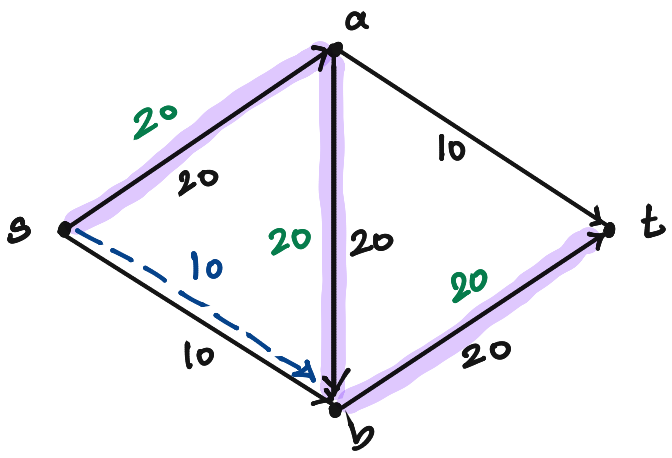
FLOW OUT OF S : 20.

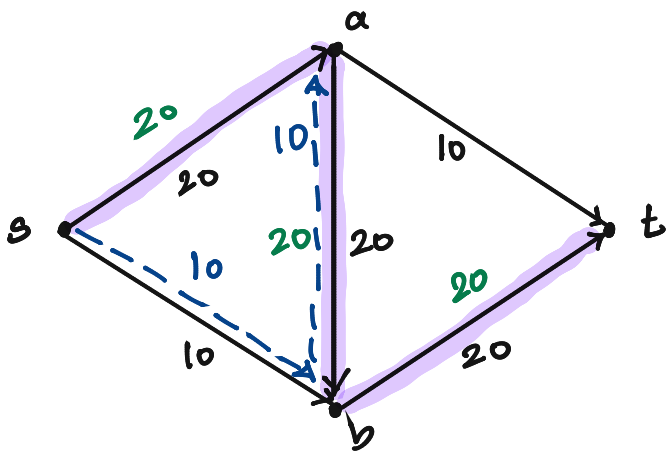
Q: Is 20 THE MAXIMUM FLOW FROM s to t?

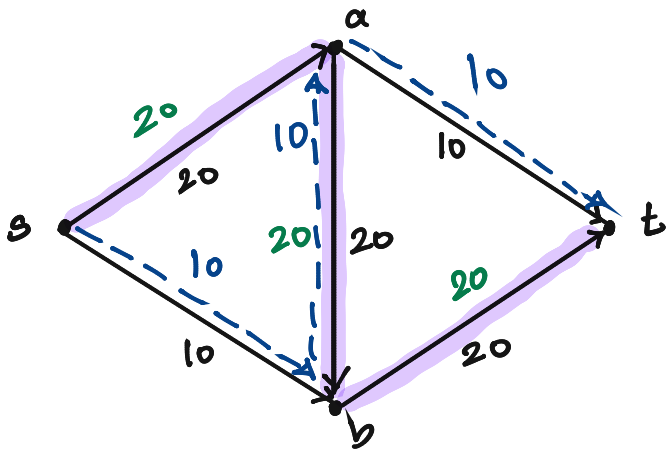


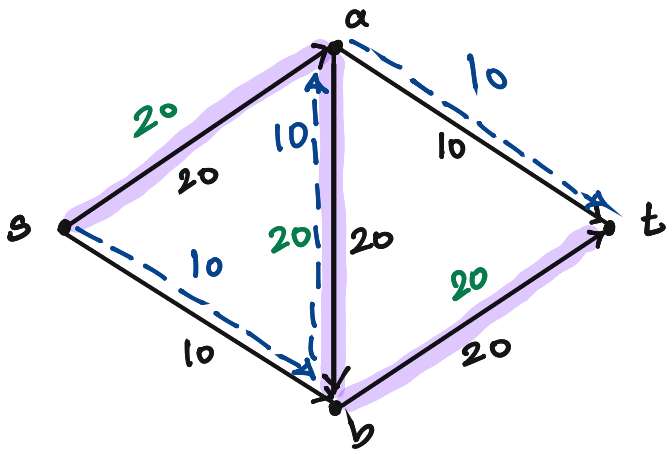
MAXIMUM FLOW = 30

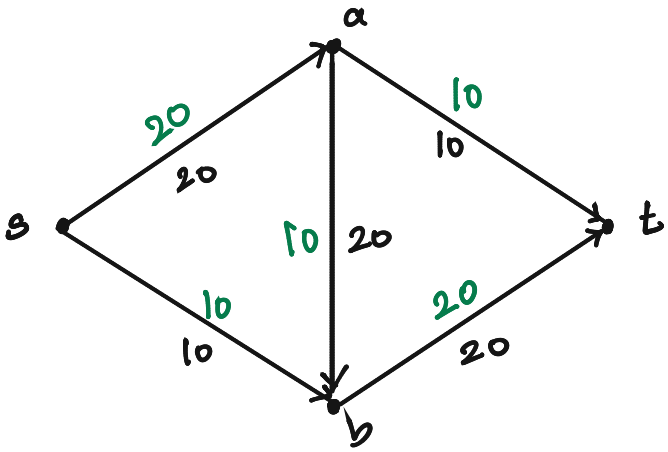

a
s
t
b
20
20
10
20
20
20
10
20

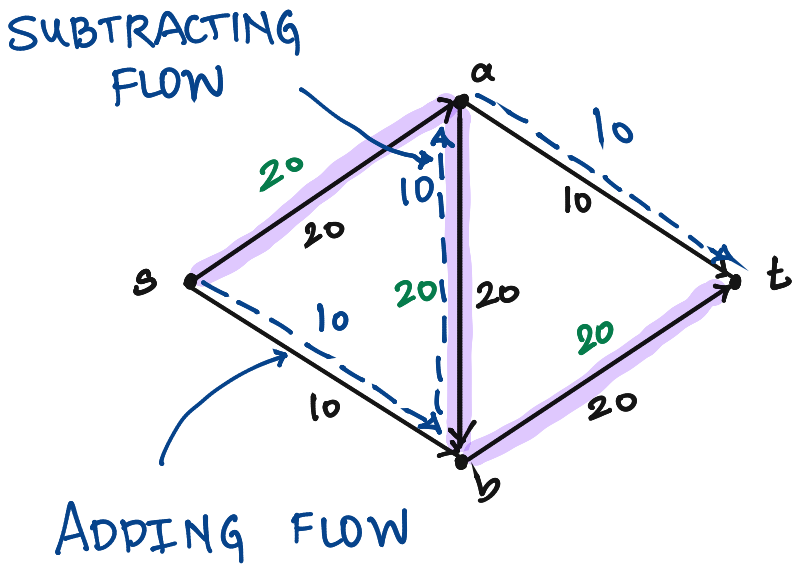

a
s
t
b
20
20
10
20
20
10
10
20
20

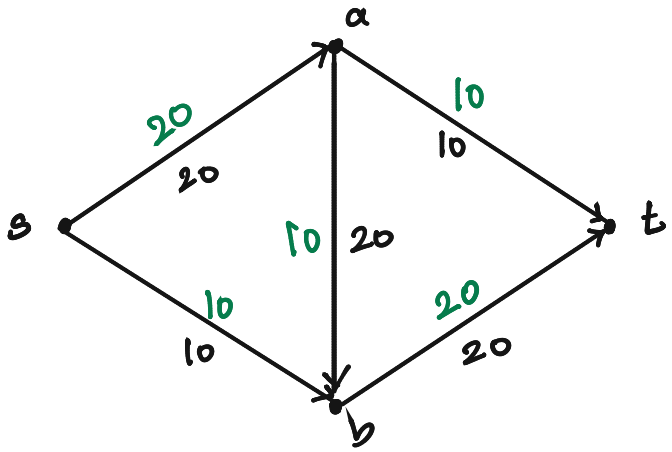

a
s
t
b
20
20
10
10
20
20
10
10
20
20


a
s
t
b
20
10
10
20
10
20
20
10
10
20
20


a
s
t
b
20
10
10
20
10
20
10
20
20
10

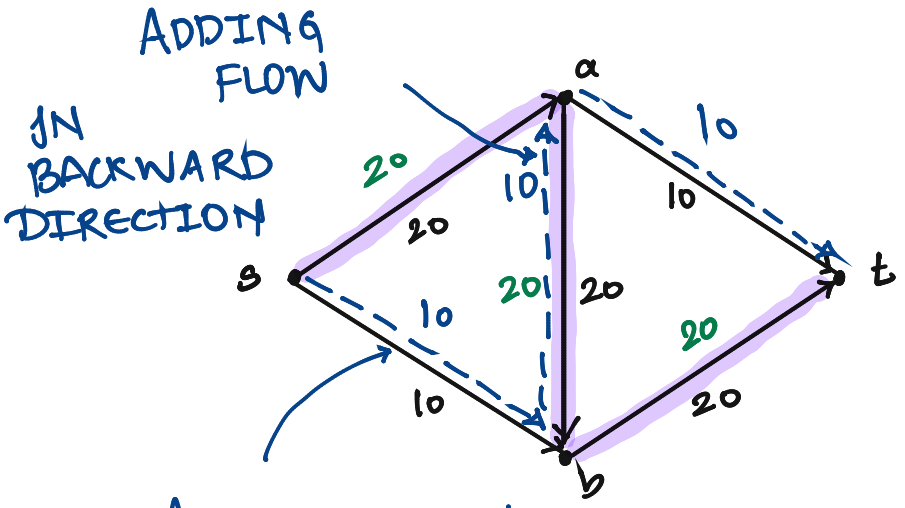

a
s
t
b
20
20
10
10
10
20
10
10
20
20



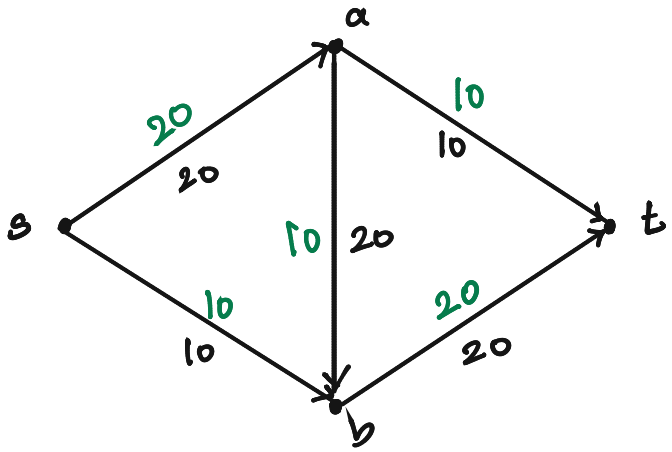


A SLIGHT CHANGE IN GREEDY APPROACH

NEEDS SOME MORE FORMALIZATION.

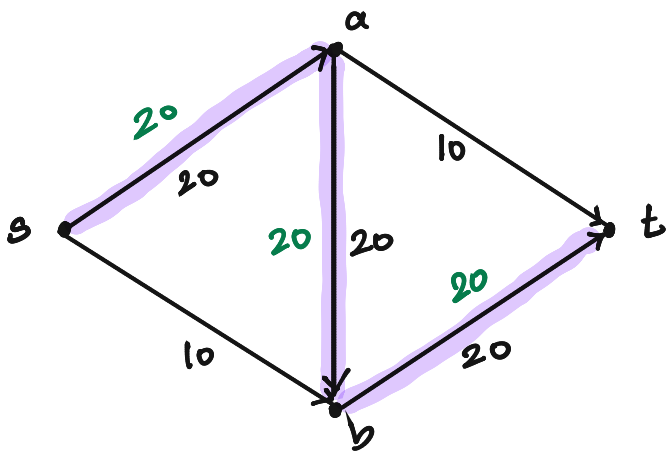


ADDING FLOW IN FORWARD DIRECTION

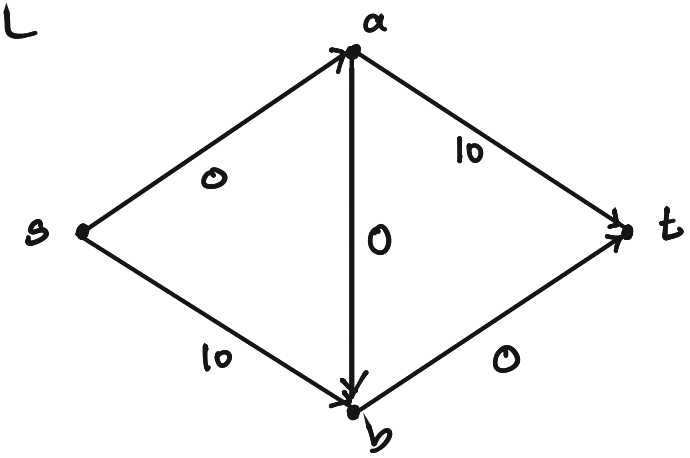


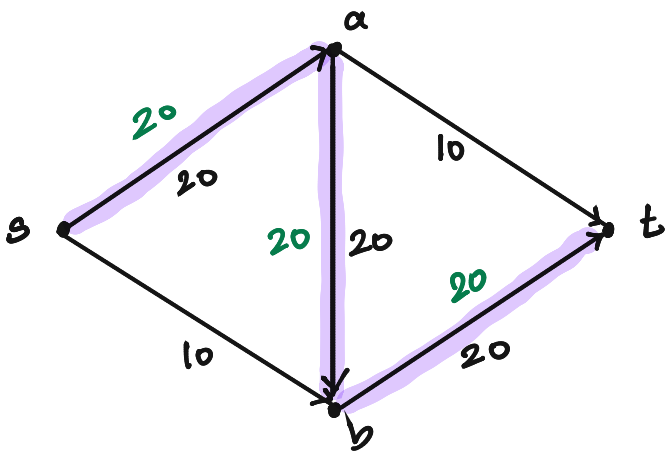
A SLIGHT CHANGE IN GREEDY APPROACH

NEEDS SOME MORE FORMALIZATION.

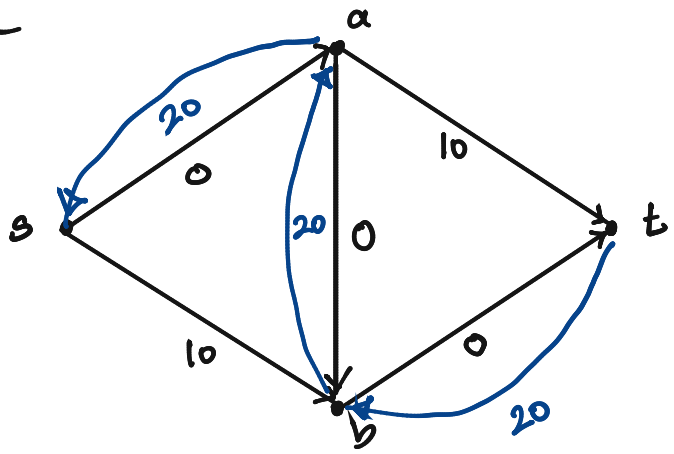


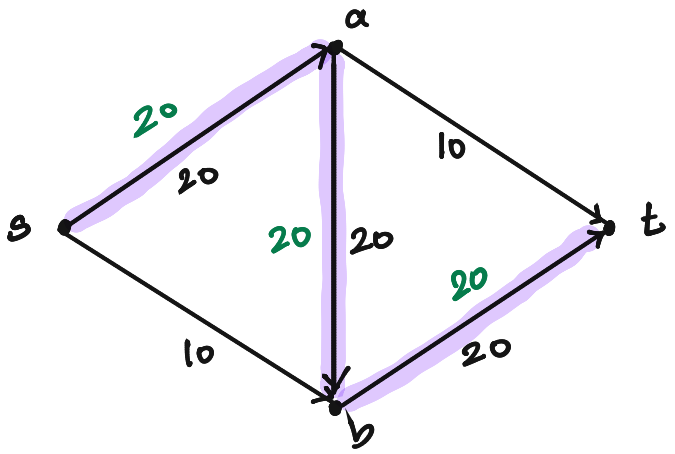
RESIDUAL GRAPH



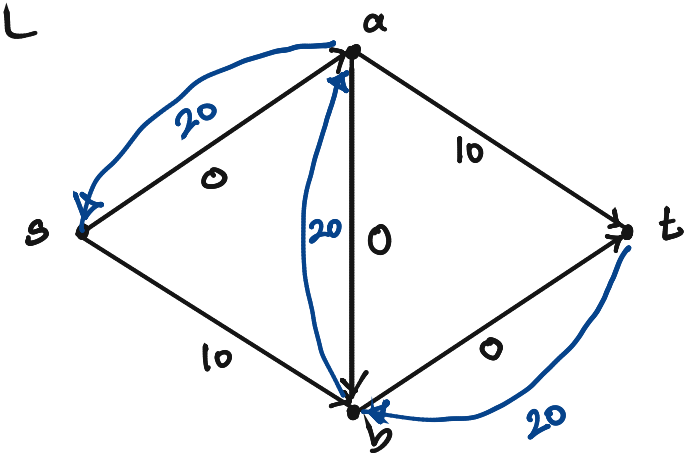


RESIDUAL GRAPH

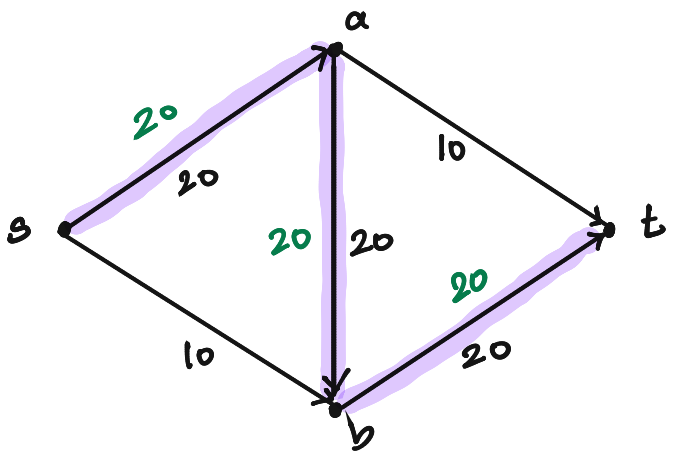




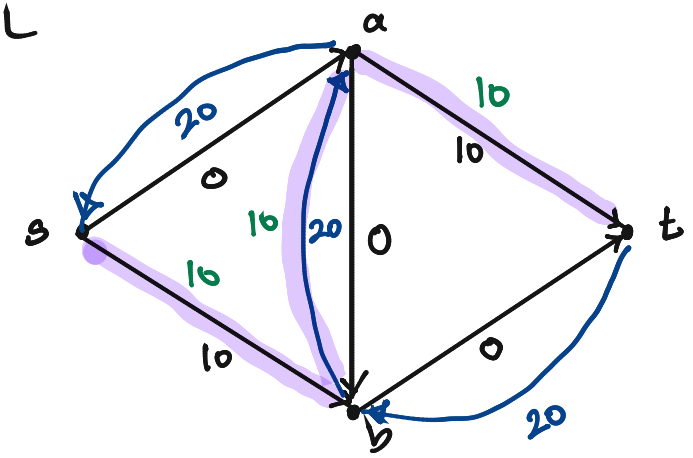
RESIDUAL GRAPH



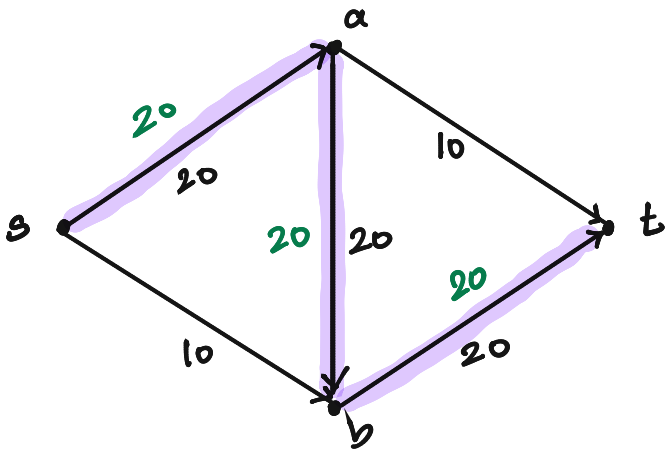
FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.



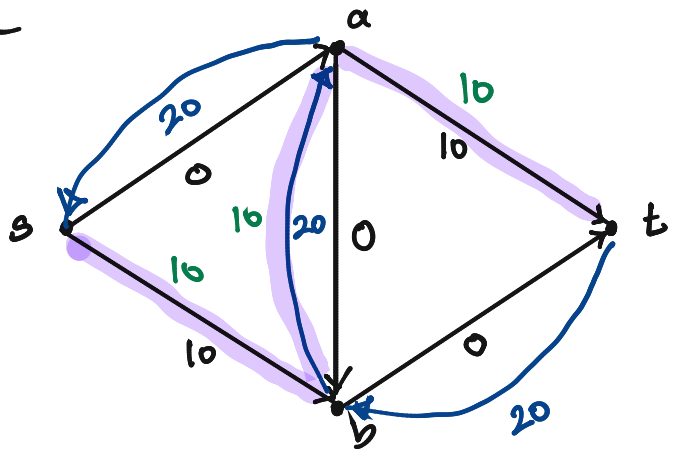
RESIDUAL GRAPH



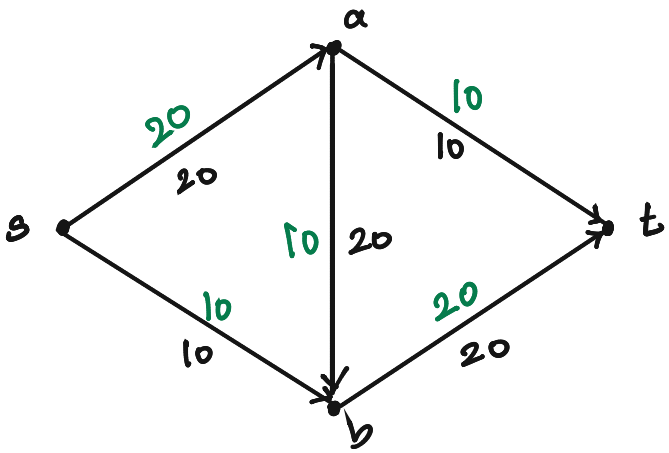
FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.

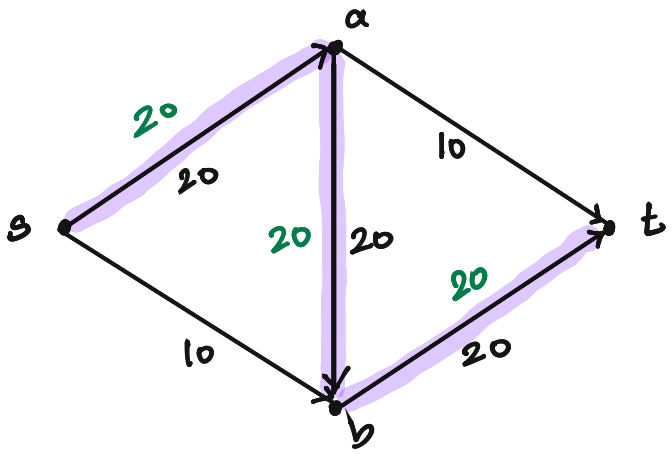


RESIDUAL GRAPH

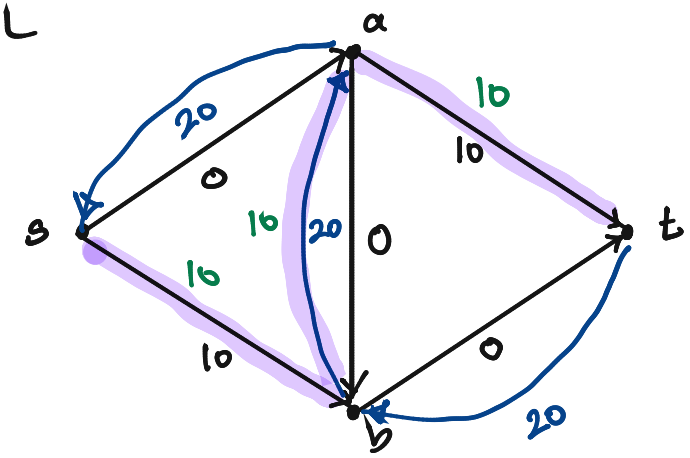


FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.

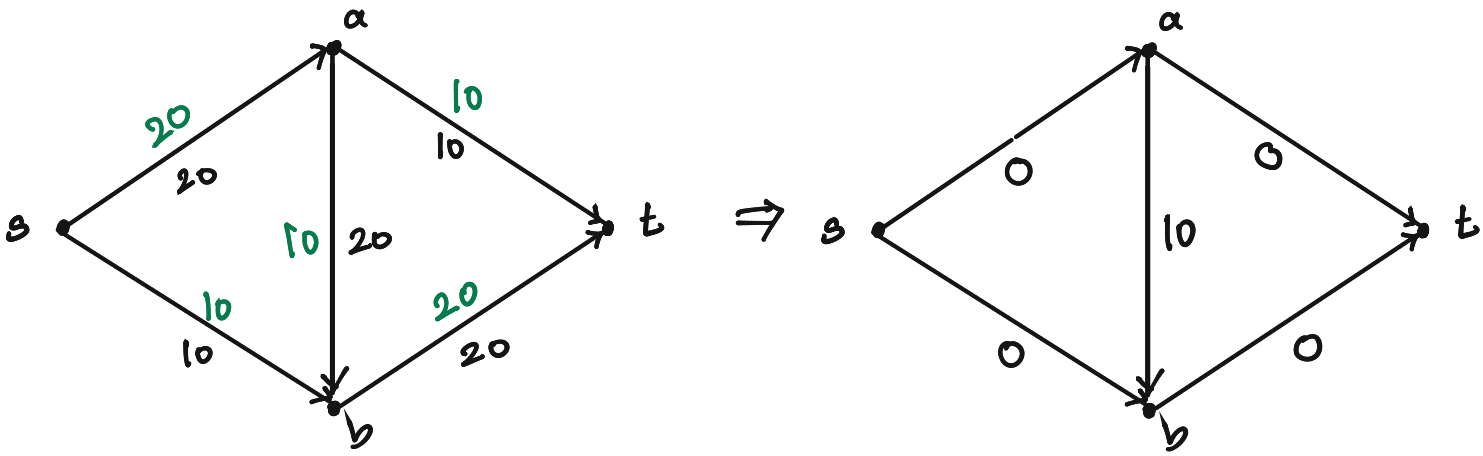


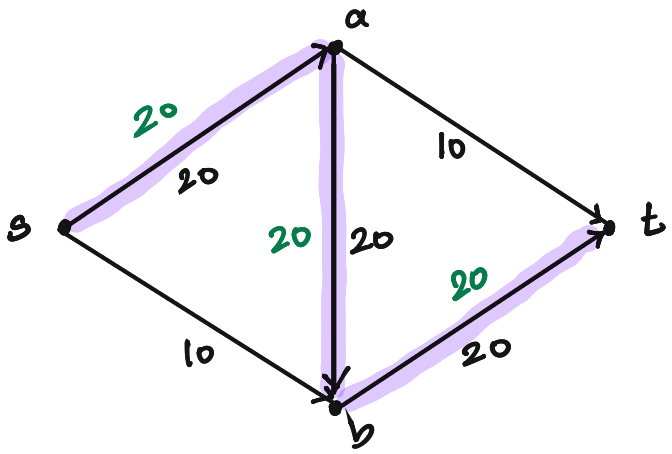


RESIDUAL GRAPH

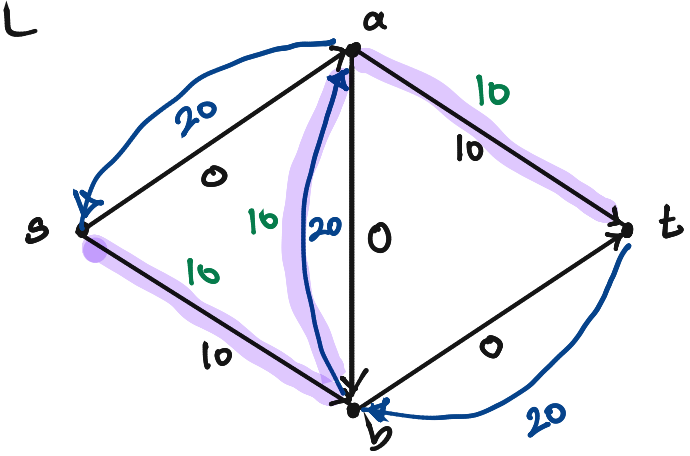


FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.

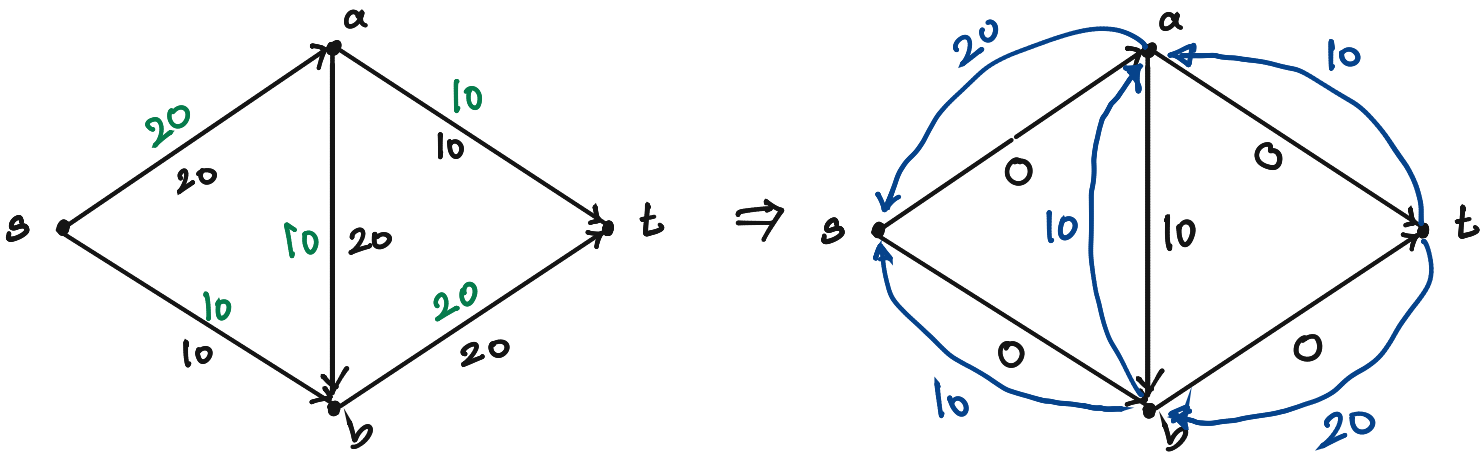


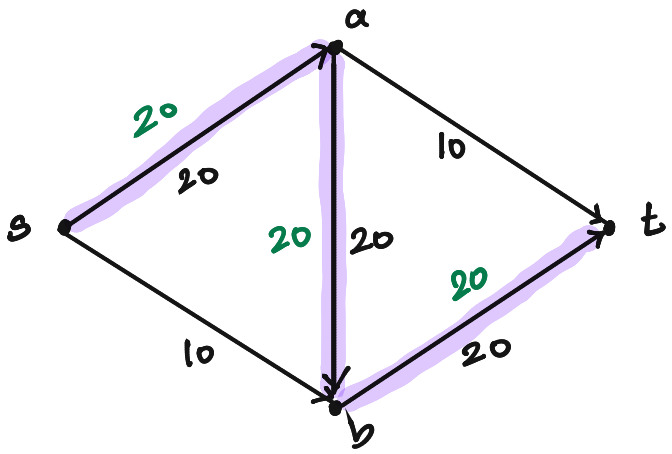


RESIDUAL GRAPH

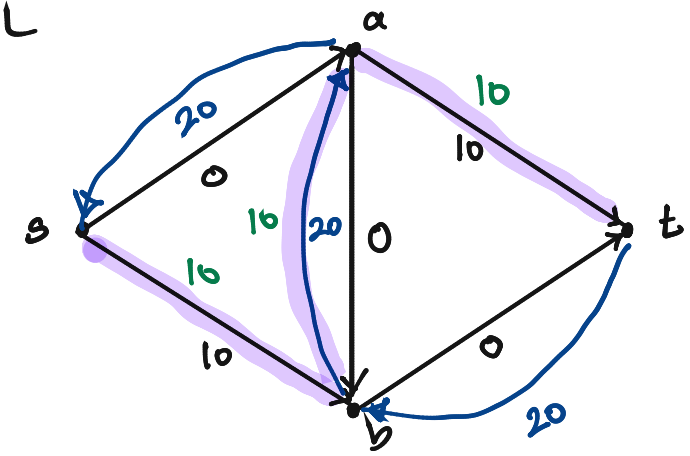


FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.

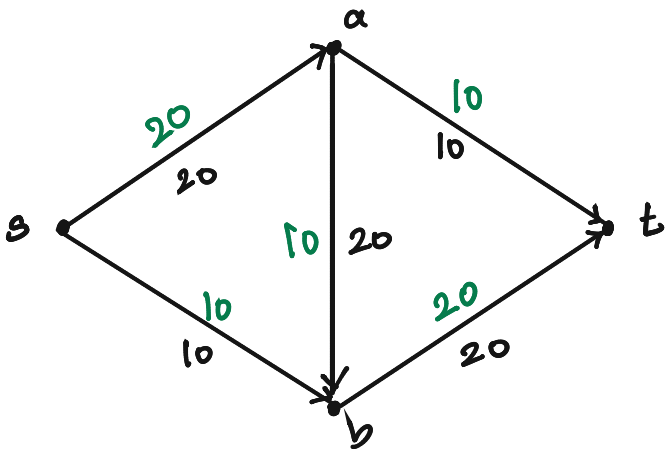




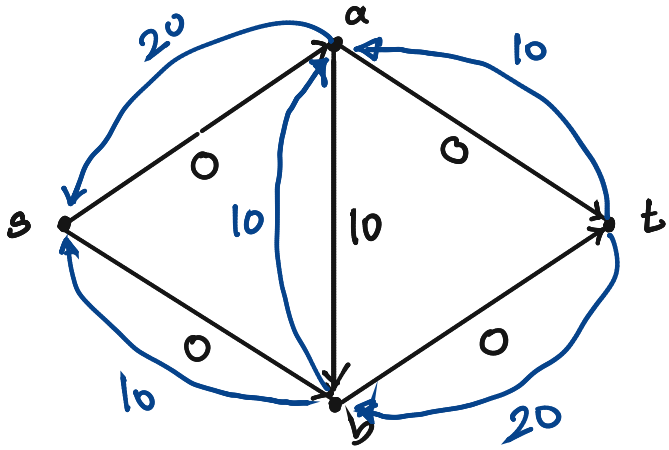
RESIDUAL GRAPH



FIND MAXIMUM BOTTLENECK PATH IN THE RESIDUAL GRAPH.



⇒



NO st PATH IN RESIDUAL GRAPH

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$c_R(e) \leftarrow$

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK : (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2)

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

③ FOR AN EDGE $e = (u, v)$

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

③ FOR AN EDGE $e = (u, v)$

(a) ADD AN EDGE $e' = (v, u)$

(b) $c_R(e') \leftarrow$

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

③ FOR AN EDGE $e = (u, v)$

(a) ADD AN EDGE $e' = (v, u)$

(b) $c_R(e') \leftarrow f(e)$

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

③ FOR AN EDGE $e = (u, v)$

(a) ADD AN EDGE $e' = (v, u)$

(b) $c_R(e') \leftarrow f(e)$

REMARK: IF $f(e) = 0$, WE MAY AS WELL DROP e' FROM G_R

Q: GIVEN A FLOW $f: E \to R^+$ (WHICH MAY NOT BE MAXIMUM), WE CAN FIND G_R IN .....

RESIDUAL GRAPH (G_R)

① VERTICES OF G_R = VERTICES OF G.

② FOR AN EDGE e,

$$c_R(e) \leftarrow c(e) - f(e)$$

REMARK: (1) $f(e) \leq c(e) \Rightarrow c_R(e) \geq 0$

(2) IF $c_R(e) = 0$, WE MAY AS WELL DROP e FROM G_R

③ FOR AN EDGE $e = (u, v)$

(a) ADD AN EDGE $e' = (v, u)$

(b) $c_R(e') \leftarrow f(e)$

REMARK: IF $f(e) = 0$, WE MAY AS WELL DROP e' FROM G_R

Q: GIVEN A FLOW $f: E \to R^+$ (WHICH MAY NOT BE MAXIMUM), WE CAN FIND G_R IN $O(m+n)$

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE (
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

            FOREACH FORWARD EDGE e in P
                f(e) ← f(e) + b ;
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

        FIND GR FOR THE NEW FLOW f
    }
}
```

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

        FIND GR FOR THE NEW FLOW f
    }
}
```

f

f'

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF:

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1)

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(a) e : FORWARD EDGE (u,v)

Q: WHAT IS CAPACITY OF e IN G_R ?

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(a) e : FORWARD EDGE (u,v)

Q: WHAT IS CAPACITY OF e IN G_R ?

A $c_R(e) \leftarrow c(e) - f(e)$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(a) e : FORWARD EDGE (u,v)

Q: WHAT IS CAPACITY OF e IN G_R ?

A $c_R(e) \leftarrow c(e) - f(e)$

Q: CAN YOU GIVE A TRIVIAL UPPER BOUND ON b ?

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(a) e : FORWARD EDGE (u,v)

Q: WHAT IS CAPACITY OF e IN G_R ?

A $c_R(e) \leftarrow c(e) - f(e)$

Q: CAN YOU GIVE A TRIVIAL UPPER BOUND ON b ?

A: $b \leq c_R(e) = c(e) - f(e)$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(a) e : FORWARD EDGE (u,v)

Q: WHAT IS CAPACITY OF e IN G_R ?

A $c_R(e) \leftarrow c(e) - f(e)$

Q: CAN YOU GIVE A TRIVIAL UPPER BOUND ON b ?

A: $b \leq c_R(e) = c(e) - f(e)$

$$f'(e) = f(e) + b \leq f(e) + (c(e) - f(e))$$

$$\Rightarrow \quad f'(e) \leq c(e)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

e: CORRESPONDING FORWARD EDGE

SINCE f IS A VALID FLOW

$f(e) \leq c(e)$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

e: CORRESPONDING FORWARD EDGE

SINCE f IS A VALID FLOW

$f(e) \leq c(e)$

AND

$f'(e) = f(e) - b$

$\Rightarrow f'(e) \leq c(e)$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

e: CORRESPONDING FORWARD EDGE

SINCE f IS A VALID FLOW

$f(e) \leq c(e)$

AND

$f'(e) = f(e) - b$

$\Rightarrow f'(e) \leq c(e)$

Q: IS $f'(e) \geq 0$?

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

e: CORRESPONDING FORWARD EDGE

SINCE f IS A VALID FLOW

$f(e) \leq c(e)$

AND

$f'(e) = f(e) - b$

$\Rightarrow f'(e) \leq c(e)$

Q: IS $f'(e) \geq 0$?

$c_R(e') = f(e)$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(1) NEED TO CHECK CAPACITY CONSTRAINTS ONLY ON EDGES OF P

(b) e': BACKWARD EDGE (u,v)

e: CORRESPONDING FORWARD EDGE

SINCE f IS A VALID FLOW

$f(e) \leq c(e)$

AND

$f'(e) = f(e) - b$

$\Rightarrow f'(e) \leq c(e)$

Q: IS $f'(e) \geq 0$?

$c_R(e') = f(e)$

$b \leq c_R(e') = f(e)$

$\Rightarrow f'(e) = f(e) - b \geq 0$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

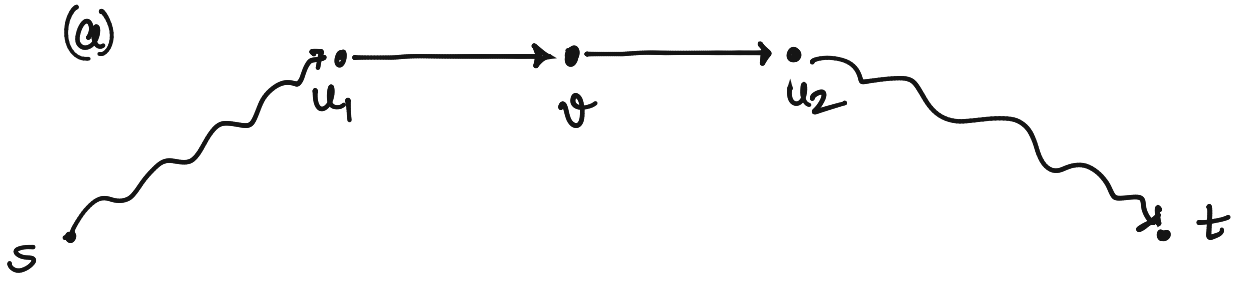
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

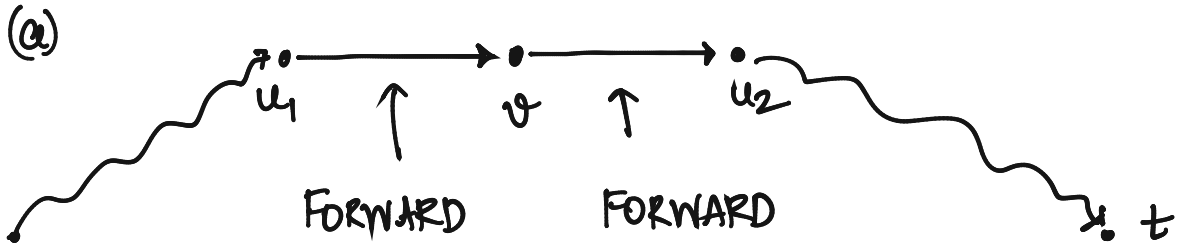
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

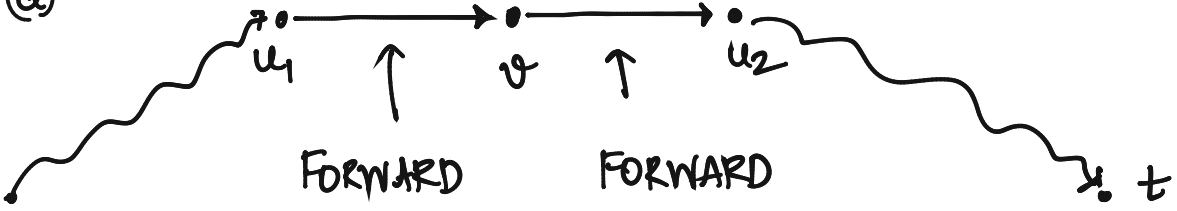
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

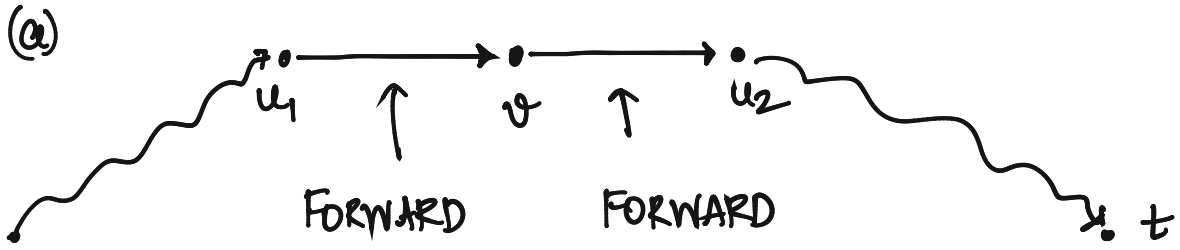
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

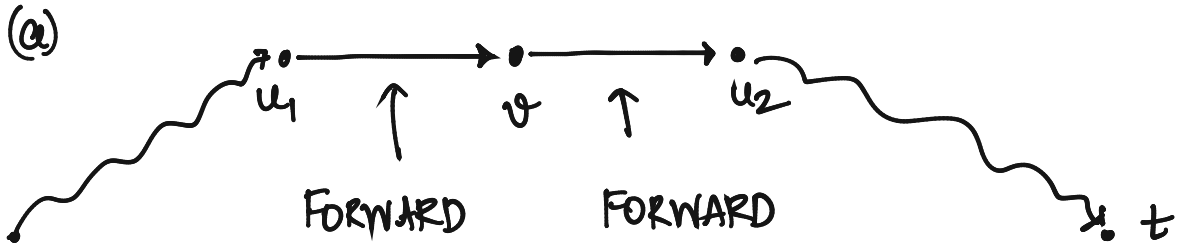
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u_1 v}} f(e) + f(u_1 v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u_1 v}} f'(e) + f(u_1 v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v)$$

$$f'(u_1 v) = f(u_1 v) + b$$

$$f'(u_2 v) = f(u_2 v) + b$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

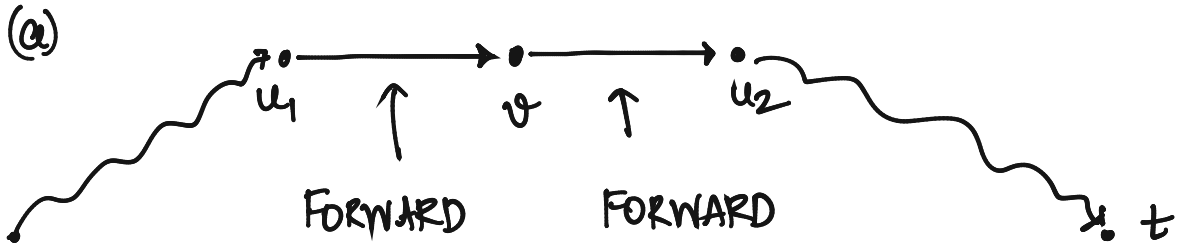
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f(u,v) + b = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v) + b$$

$$f'(u,v) = f(u_1 v) + b$$

$$f'(u_2 v) = f(u_2 v) + b$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

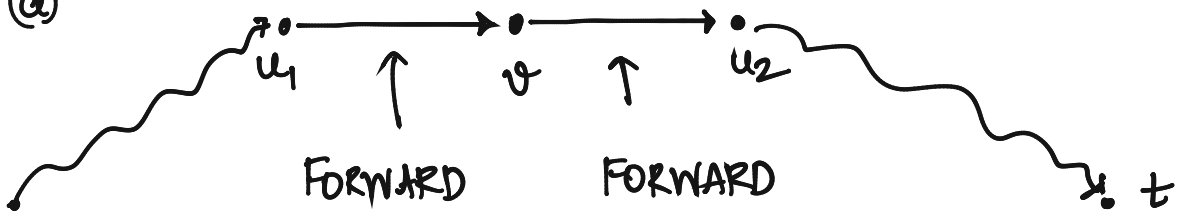
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f(u,v) + b = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v) + b$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f'(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f'(u_2 v)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

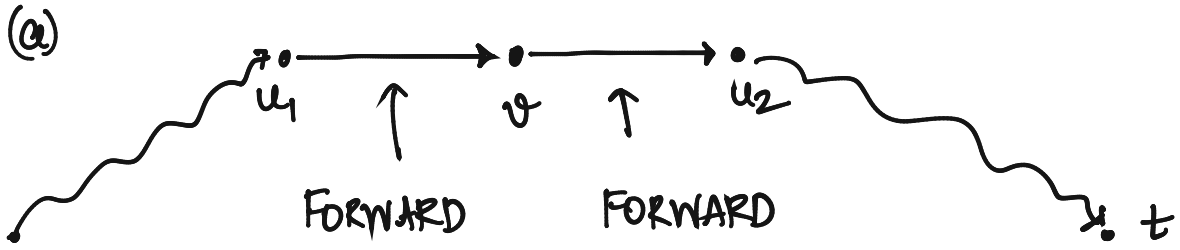
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f(u,v) + b = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v) + b$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f'(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f'(u_2 v)$$

$$\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

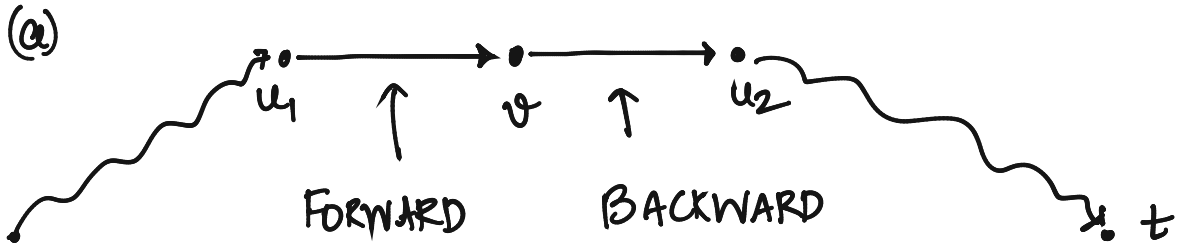
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f(e) + f(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f(e) + f(u_2 v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f(u,v) + b = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f(u_2 v) + b$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except } u,v}} f'(e) + f'(u,v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2 v}} f'(e) + f'(u_2 v)$$

$$\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$$

LEMMA: IF f IS A VALID FLOW, THEN f' IS ALSO A VALID FLOW.

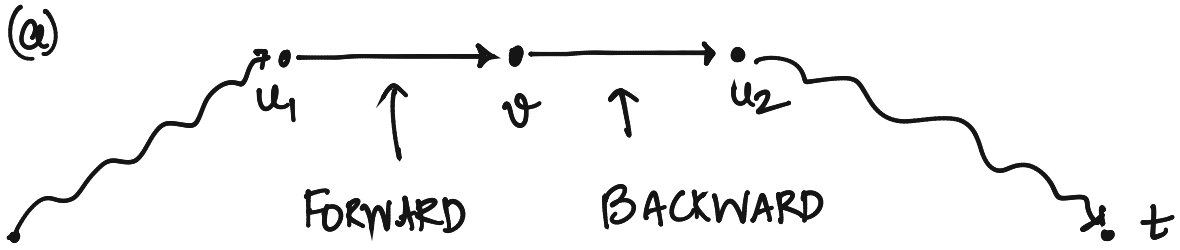
PROOF: TO SHOW

(1) $f'(e) \leq c(e) \quad \forall e \in E$ (CAPACITY CONSTRAINT)

(2) $\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$

(3) FLOW CONSERVATION

(a)



$$\sum_{e \text{ IN } v} f(e) = \sum_{e \text{ OUT OF } v} f(e)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except} \\ u_1v \text{ \& } u_2v}} f(e) + f(u_1v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except}}} f(e)$$

$$f(u_2v)$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except} \\ u_1v \text{ \& } u_2v}} f'(e) + f(u_1v) + b = \sum_{\substack{e \text{ OUT OF } v \\ \text{except}}} f'(e)$$

$$+ f(u_2v) - b$$

$$\sum_{\substack{e \text{ IN } v \\ \text{except} \\ u_1v \text{ \& } u_2v}} f'(e) + f'(u_1v) = \sum_{\substack{e \text{ OUT OF } v \\ \text{except } u_2v}} f'(e)$$

$$+ f'(u_2v)$$

$$\sum_{e \text{ IN } v} f'(e) = \sum_{e \text{ OUT OF } v} f'(e)$$

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← MAXIMUM BOTTLENECK PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

        FIND GR FOR THE NEW FLOW f
    }
}
```

MAX-FLOW (G) ALL CAPCITIES ARE INTEGERS.

```
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← ANY st PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

        FIND GR FOR THE NEW FLOW f
    }
}
```

LEMMA: AFTER EACH ITERATION OF WHILE LOOP, $f(e)$ IS AN INTEGER $\forall e \in E$.

LEMMA: LET f BE A FLOW AT THE START OF WHILE LOOP & f' AT THE END OF WHILE LOOP, THEN

$$\sum_{e \text{ out of } s} f'(e) > \sum_{e \text{ out of } s} f(e)$$

LEMMA: LET f BE A FLOW AT THE START OF WHILE LOOP & f' AT THE END OF WHILE LOOP, THEN

$$\sum_{e \text{ out of } s} f'(e) > \sum_{e \text{ out of } s} f(e)$$

FLOW OUT OF S IN f' > FLOW OUT OF S IN f.

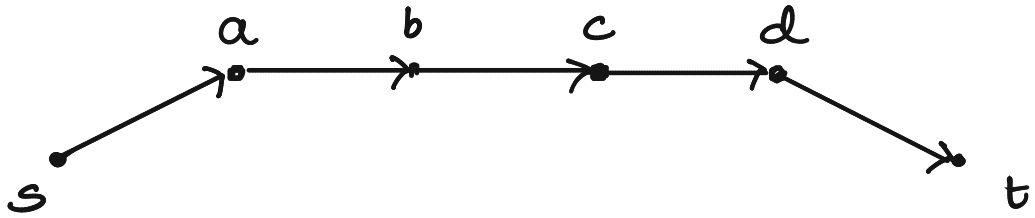
PROOF:

LEMMA: LET f BE A FLOW AT THE START OF WHILE LOOP & f' AT THE END OF WHILE LOOP, THEN

$$\sum_{e \text{ OUT OF } s} f'(e) > \sum_{e \text{ OUT of } s} f(e)$$

FLOW OUT OF S IN f' > FLOW OUT OF S IN f.

PROOF:



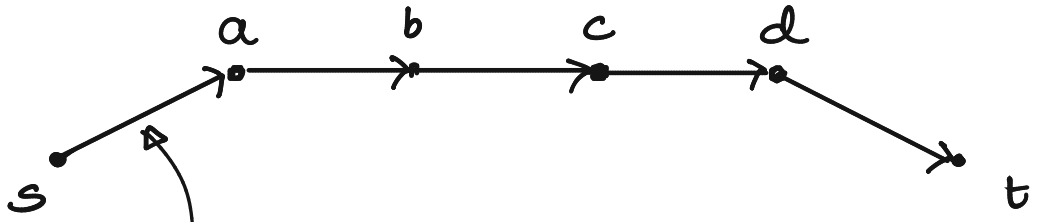
SIMPLE PATH $\Rightarrow$ EACH VERTEX IS VISITED ONCE.

LEMMA: LET f BE A FLOW AT THE START OF WHILE LOOP & f' AT THE END OF WHILE LOOP, THEN

$$\sum_{e \text{ OUT OF } s} f'(e) > \sum_{e \text{ OUT of } s} f(e)$$

FLOW OUT OF s IN f' > FLOW OUT OF s IN f.

PROOF:



SIMPLE PATH $\Rightarrow$ EACH VERTEX IS VISITED ONCE.

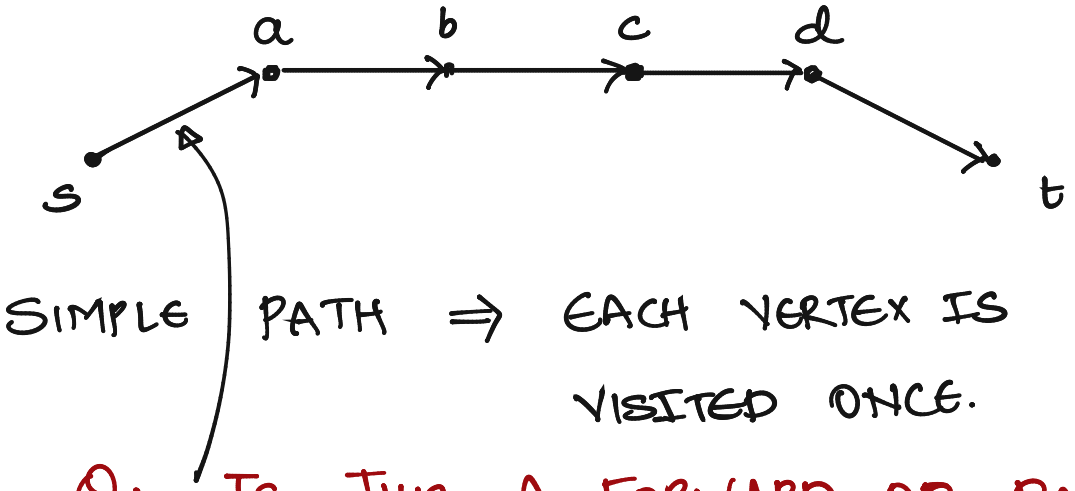
Q: IS THIS A FORWARD OR BACKWARD EDGE ?

LEMMA: LET f BE A FLOW AT THE START OF WHILE LOOP & f' AT THE END OF WHILE LOOP, THEN

$$\sum_{e \text{ OUT OF } s} f'(e) > \sum_{e \text{ OUT of } s} f(e)$$

FLOW OUT OF s IN f' > FLOW OUT OF s IN f.

PROOF:



SIMPLE PATH ⇒ EACH VERTEX IS VISITED ONCE.

Q: IS THIS A FORWARD OR BACKWARD EDGE ?

A FORWARD.

$$\Rightarrow \sum_{e \text{ OUT OF } s} f'(e) = \sum_{e \text{ OUT of } s} f(e) + b$$

$$\Rightarrow \text{FLOW OUT of } s \text{ IN } f' > \text{FLOW OUT OF } s \text{ IN } f.$$

OBSERVATION: FLOW OUT OF S INCREASES BY ATLEAST 1 AFTER EACH ITERATION

OBSERVATION: FLOW OUT OF s INCREASES BY ATLEAST 1 AFTER EACH ITERATION

Q: WHAT IS THE MAXIMUM FLOW FROM s to t ?

OBSERVATION: FLOW OUT OF s INCREASES BY ATLEAST 1 AFTER EACH ITERATION

Q: WHAT IS THE MAXIMUM FLOW FROM s to t ?

A: $C = \sum_{\substack{e \text{ out} \\ \text{of } s}} c(e)$

OBSERVATION: FLOW OUT OF s INCREASES BY ATLEAST 1 AFTER EACH ITERATION

Q: WHAT IS THE MAXIMUM FLOW FROM s to t ?

A: $C = \sum_{\substack{e \text{ out} \\ \text{of } s}} c(e)$

OUR WHILE LOOP RUNS ATMOST _____ ITERATIONS

OBSERVATION: FLOW OUT OF s INCREASES BY ATLEAST 1 AFTER EACH ITERATION

Q: WHAT IS THE MAXIMUM FLOW FROM s to t ?

A: $C = \sum_{\substack{e \text{ out} \\ \text{of } s}} c(e)$

OUR WHILE LOOP RUNS ATMOST C. ITERATIONS

FORD-FULKERSON ALGORITHM.

```
MAX-FLOW (G)
{   f ← INITIAL FLOW
    GR ← INITIAL RESIDUAL GRAPH.

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← ANY st PATH IN GR
        b ← MINIMUM CAPACITY EDGE IN P.

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

        FIND GR FOR THE NEW FLOW f
    }
}
```

FORD-FULKERSON ALGORITHM.

```
MAX-FLOW (G)
{   f ← INITIAL FLOW                          — O(m)
    GR ← INITIAL RESIDUAL GRAPH.              — O(m+n)

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← ANY st PATH IN GR                 — O(m+n)
        b ← MINIMUM CAPACITY EDGE IN P.       O(n)

O(n)    FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

        FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

O(m+n)  { FIND GR FOR THE NEW FLOW f
        }
}
```

FORD-FULKERSON ALGORITHM.

```
MAX-FLOW (G)
{   f ← INITIAL FLOW                          — O(m)
    GR ← INITIAL RESIDUAL GRAPH.              — O(m+n)

    WHILE ( THERE EXISTS A st PATH IN GR)
    {   P ← ANY st PATH IN GR                 — O(m+n)
        b ← MINIMUM CAPACITY EDGE IN P.       O(n)

        FOREACH FORWARD EDGE e in P
            f(e) ← f(e) + b ;

O(n)    FOREACH BACKWARD EDGE e' in P
        {   e ← CORRESPONDING FORWARD EDGE
            f(e) ← f(e) - b
        }

O(m+n)  { FIND GR FOR THE NEW FLOW f
        }
}
```

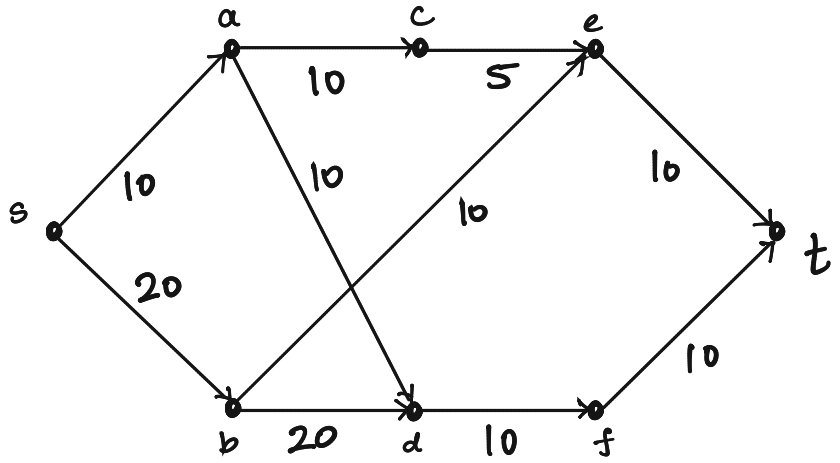
RUNNING TIME : $O((m+n)C)$.

$= O(mC)$

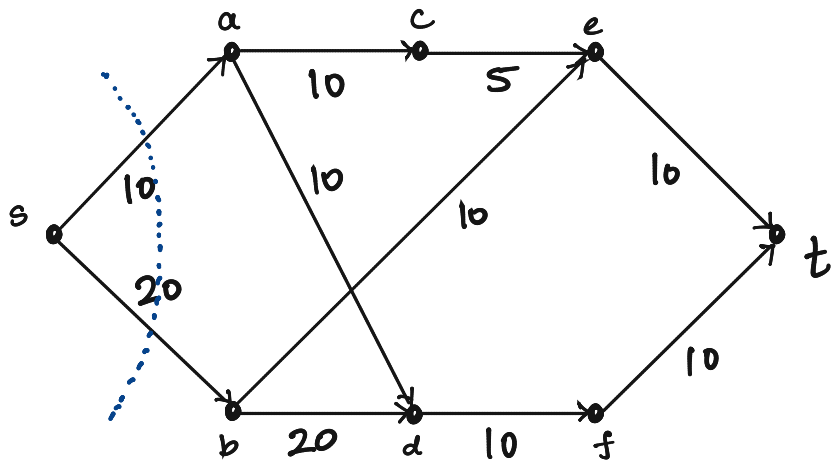
SHOWN THAT THE ALGORITHM TERMINATES AND RETURNS A VALID FLOW IN $O(mC)$ TIME.

FOR CORRECTNESS, WE STILL NEED TO SHOW THAT WE RETURN A MAXIMUM FLOW.

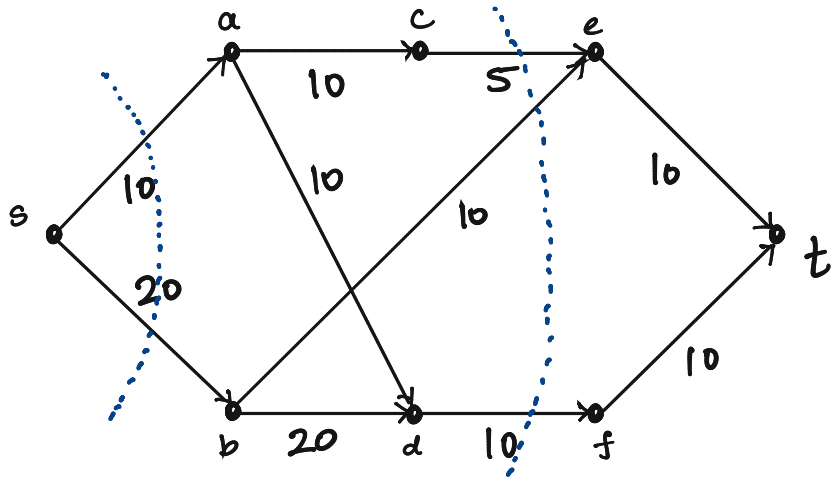
IN FACT THIS ALGORITHM IS FAMOUS DUE TO ITS CORRECTNESS ANALYSIS.



FLOW FROM s to t IS ALWAYS ≤

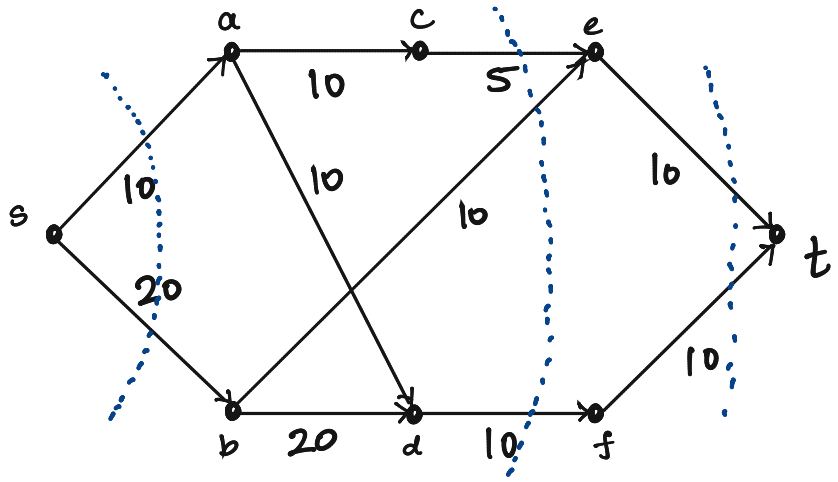


FLOW FROM s to t IS ALWAYS $\leq$ 30.



FLOW FROM s to t IS ALWAYS $\leq$ 30.

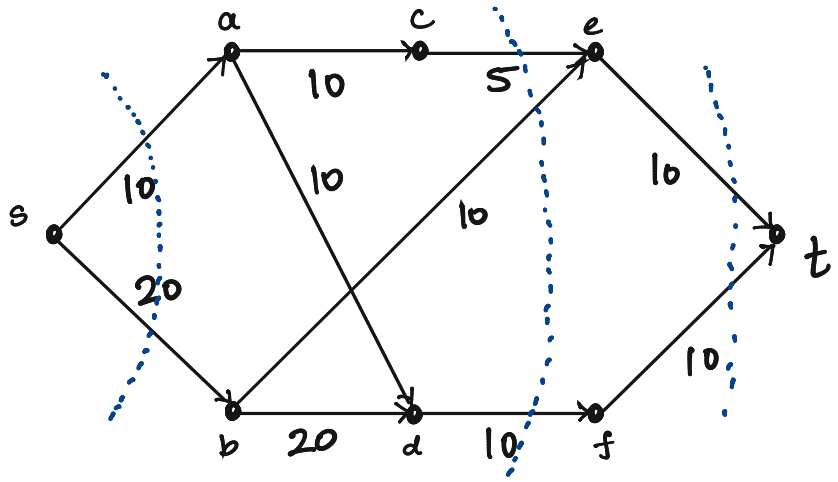
FLOW FROM s to t $\leq$ 25



FLOW FROM s to t IS ALWAYS ≤ 30.

FLOW FROM s to t ≤ 25

FLOW FROM s to t ≤ 20

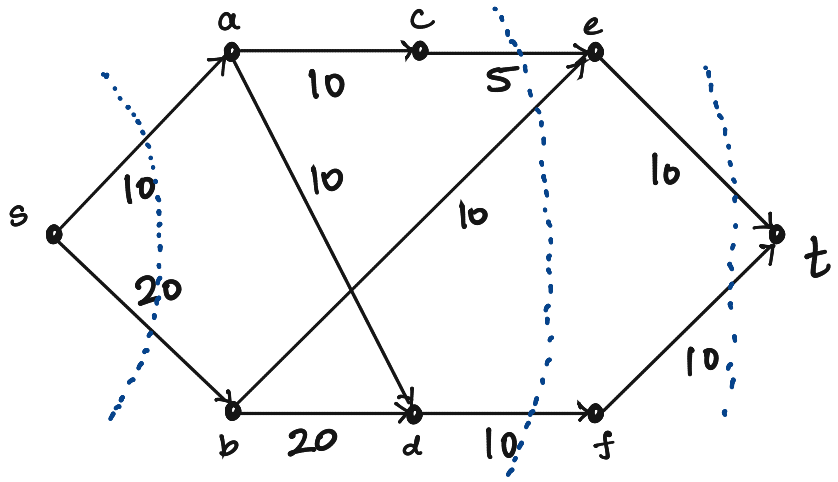


FLOW FROM s to t IS ALWAYS $\leq$ 30.

$A = \{s\}$

$B = \{a, b, c, d, e, f, t\}$

$c(A,B) = \{(s,a), (s,b)\}$



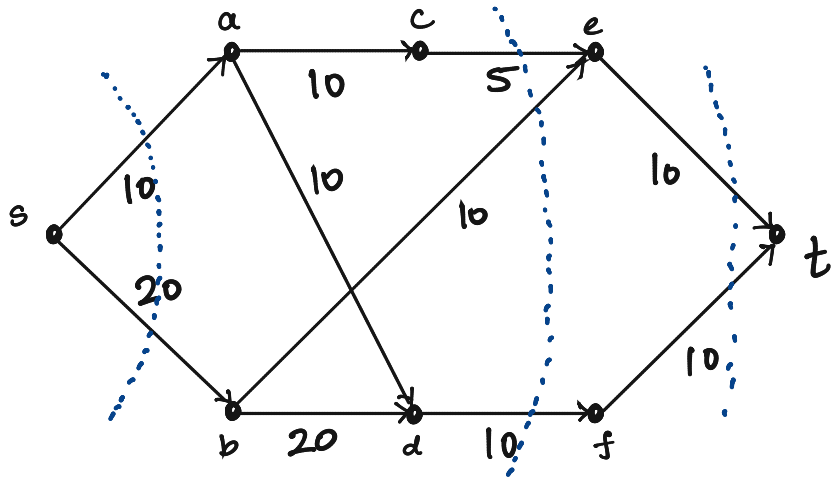
FLOW FROM s to t IS ALWAYS $\leq$ 30.

FLOW FROM s to t $\leq$ 25

$A = \{s, a, b, c, d\}$

$B = \{e, f, t\}$

$c(A, B) = \{(c, e), (d, f), (b, e)\}$



FLOW FROM s to t IS ALWAYS $\leq$ 30.

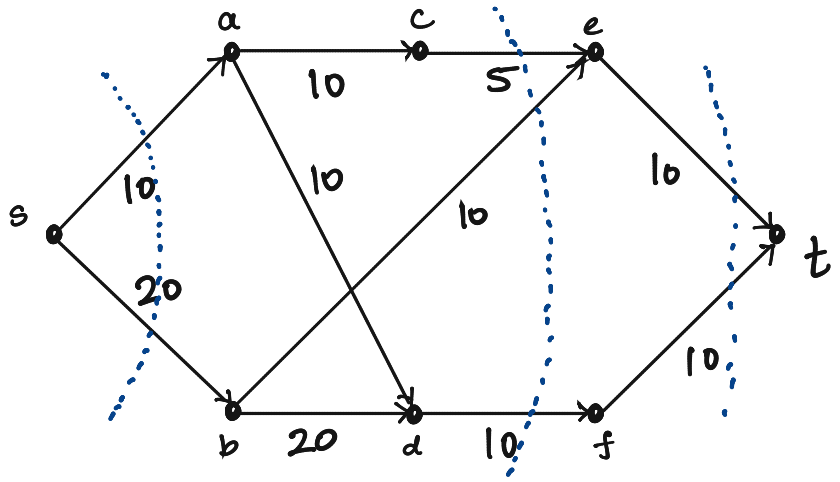
FLOW FROM s to t $\leq$ 25

FLOW FROM s to t $\leq$ 20

$A = \{ s, a, b, c, d, e, f \}$

$B = \{ t \}$

$c(A,B) = \{ (e,t), (f,t) \}$.



FLOW FROM s to t IS ALWAYS $\leq$ 30.

FLOW FROM s to t $\leq$ 25

FLOW FROM s to t $\leq$ 20

A = SET CONTAINING s

B = SET CONTAINING t

$C(A,B) = \{(u,v) \mid u \in A \text{ & } v \in B\}$

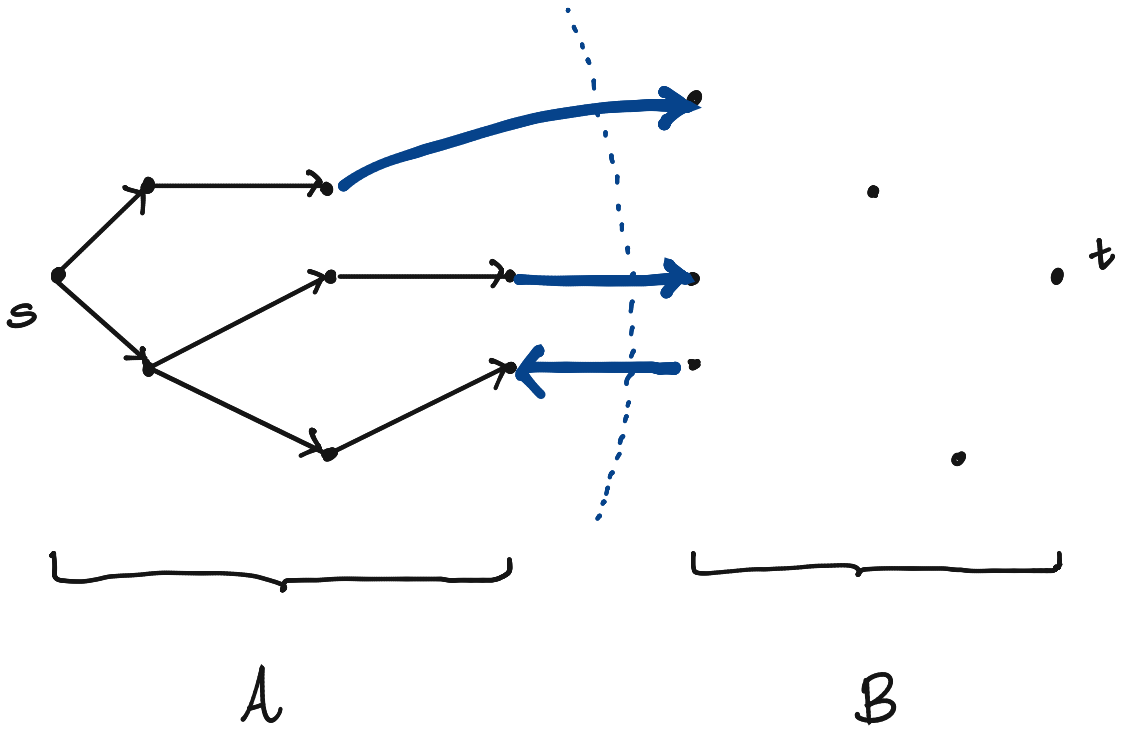
LEMMA: GIVEN A CUT $c(A,B)$, THE FLOW FROM s to t $\leq \sum_{e \in C(A,B)} w(e)$

LEMMA : LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

$$\text{FLOW FROM } s \text{ to } t = \sum_{\substack{e \text{ OUT} \\ \text{of } A}} f(e) - \sum_{\substack{e \text{ IN} \\ A}} f(e)$$

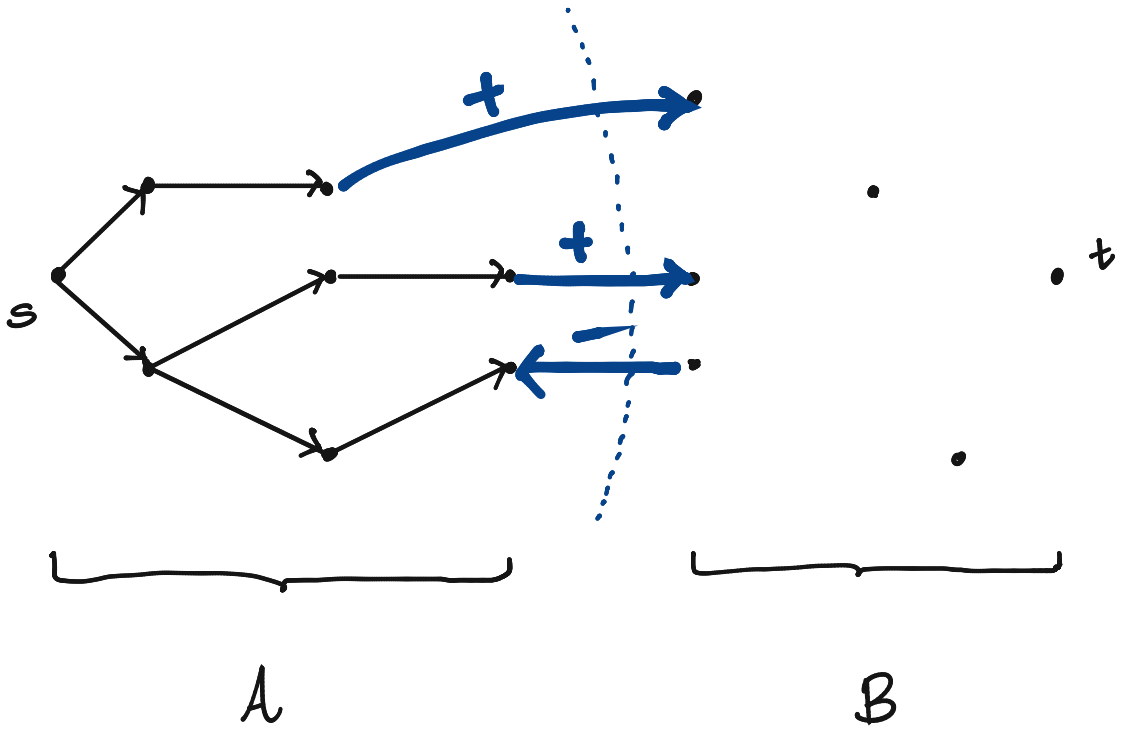
LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

$$\text{FLOW FROM } s \text{ to } t = \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$$



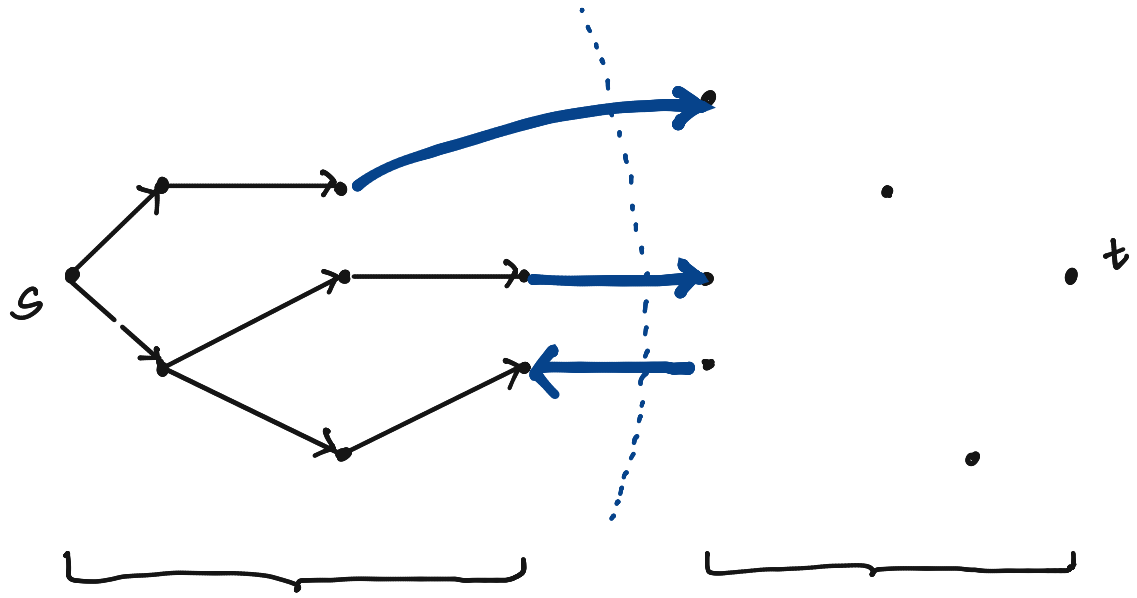
LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

$$\text{FLOW FROM } s \text{ to } t = \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$$



LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

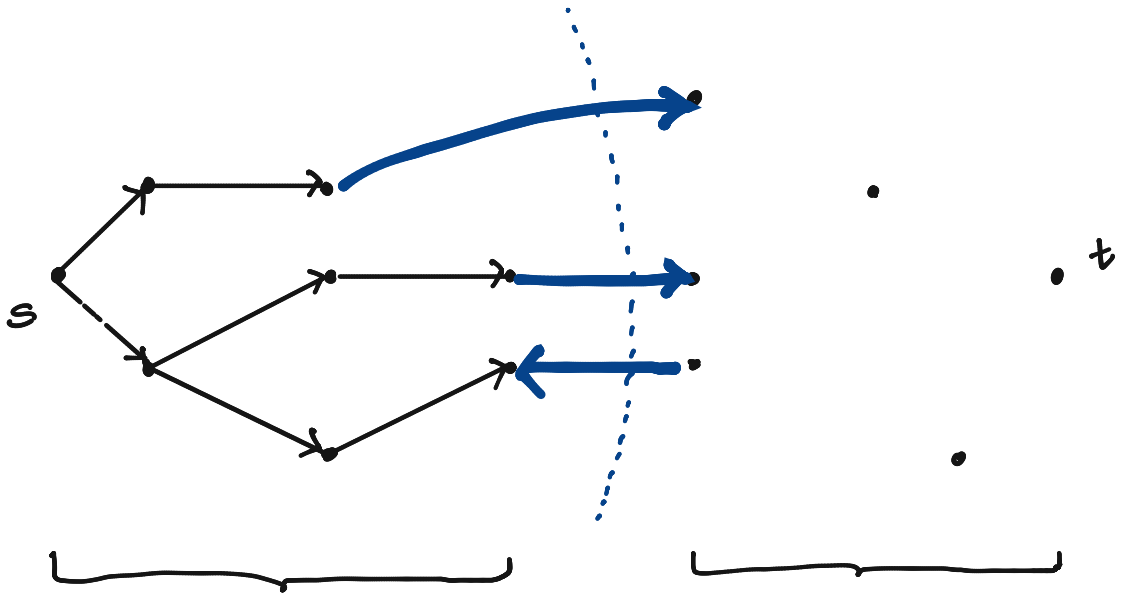
FLOW FROM s to t $= \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$



$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e)$$

LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

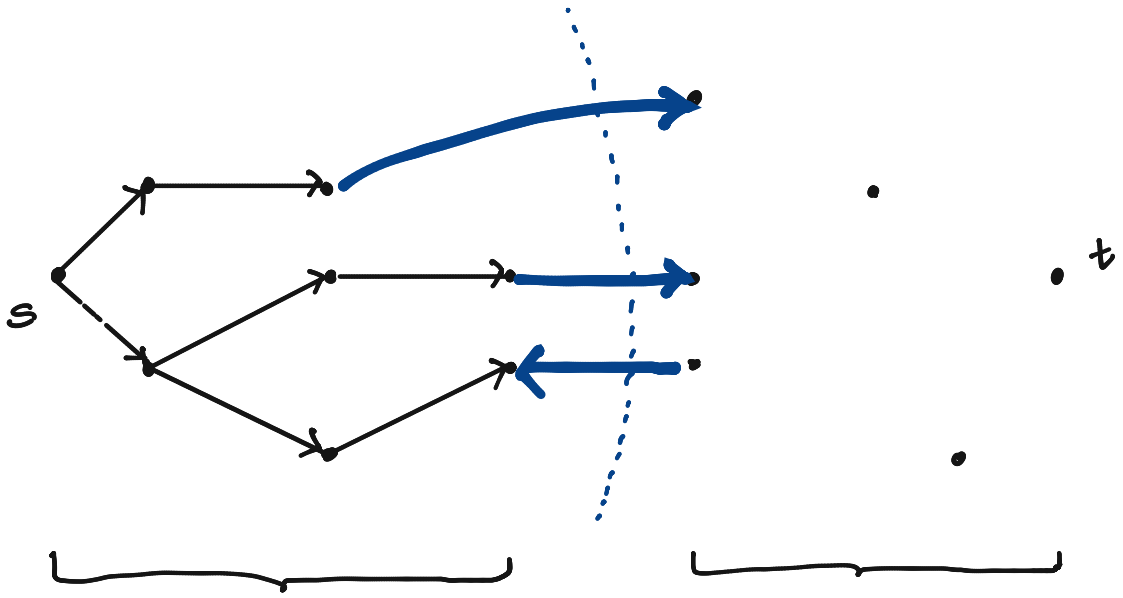
$$\text{FLOW FROM } s \text{ to } t = \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$$



$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e) = \text{FLOW FROM } s \text{ to } t$$

LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

FLOW FROM s to t $= \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$



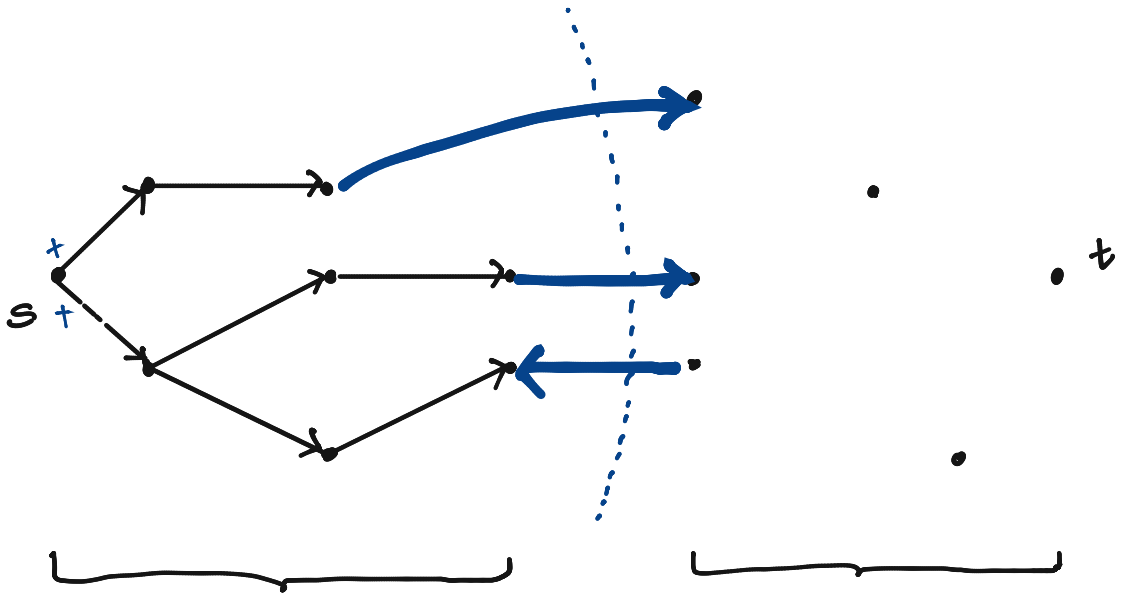
$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e) = \text{FLOW FROM } s \text{ to } t$$

FOR ANY OTHER INTERNAL NODE v IN A,

$$\sum_{e \text{ OUT OF } v} f(e) - \sum_{e \text{ IN } v} f(e)$$

LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

FLOW FROM s to t $= \sum_{\substack{e \text{ OUT} \\ \text{of } A}} f(e) - \sum_{\substack{e \text{ IN} \\ A}} f(e)$



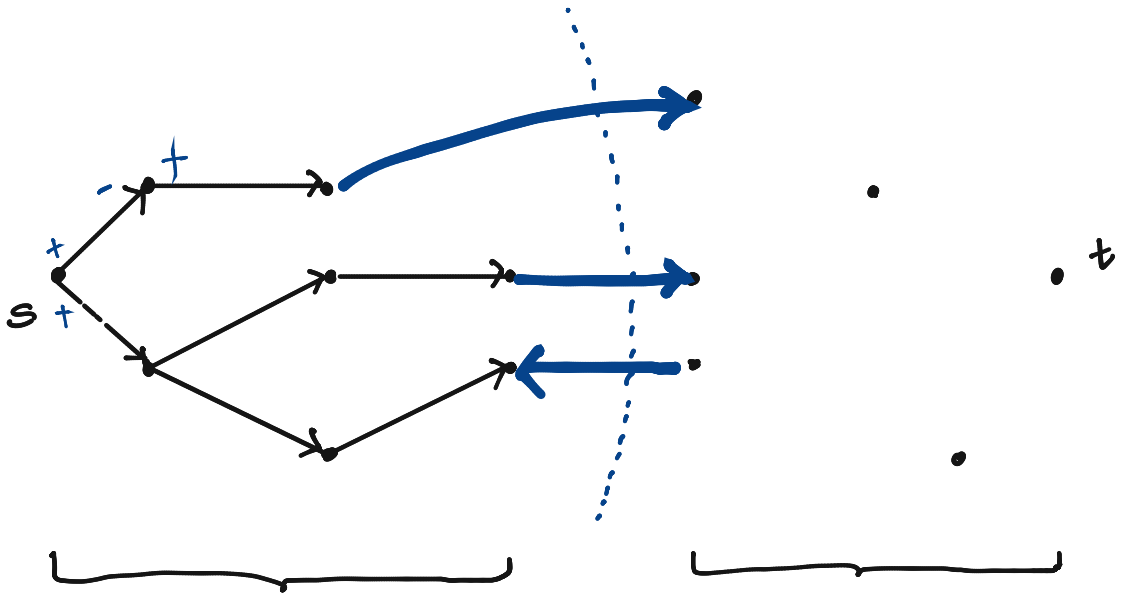
$$\sum_{\substack{e \text{ OUT} \\ \text{OF } s}} f(e) - \sum_{\substack{e \text{ IN} \\ s}} f(e) = \text{FLOW FROM } s \text{ to } t$$

FOR ANY OTHER INTERNAL NODE v IN A,

$$\sum_{\substack{e \text{ OUT} \\ \text{OF } v}} f(e) - \sum_{\substack{e \text{ IN} \\ v}} f(e) = 0$$

LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

FLOW FROM s to t $= \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$



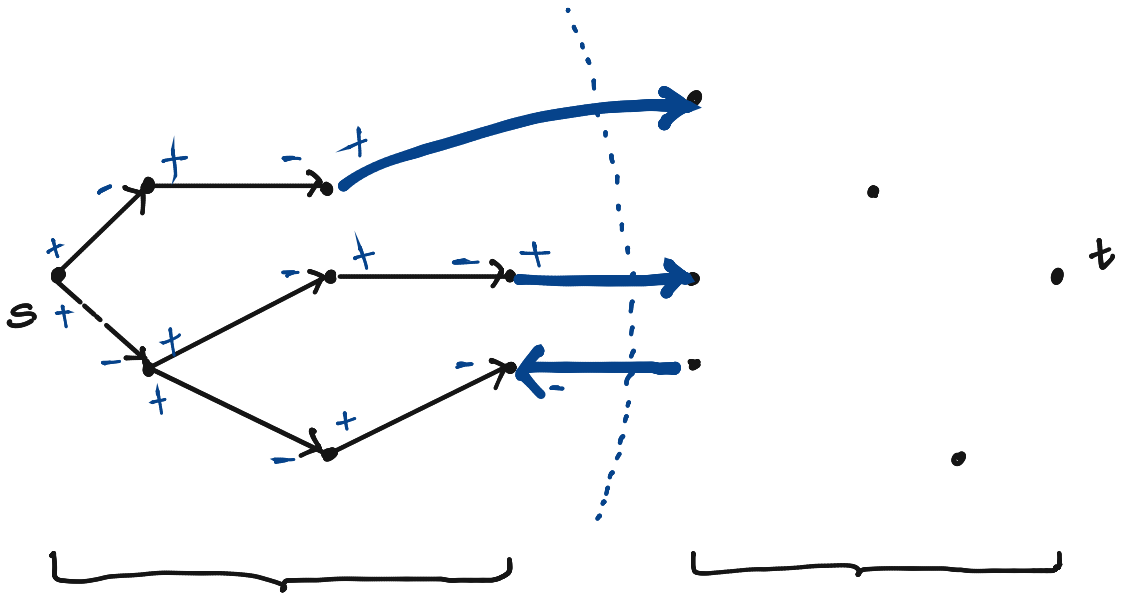
$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e) = \text{FLOW FROM } s \text{ to } t$$

FOR ANY OTHER INTERNAL NODE v IN A,

$$\sum_{e \text{ OUT OF } v} f(e) - \sum_{e \text{ IN } v} f(e) = 0$$

LEMMA : LET f BE A st FLOW AND $c(A,B)$ BE St CUT, THEN

FLOW FROM s to t $= \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$



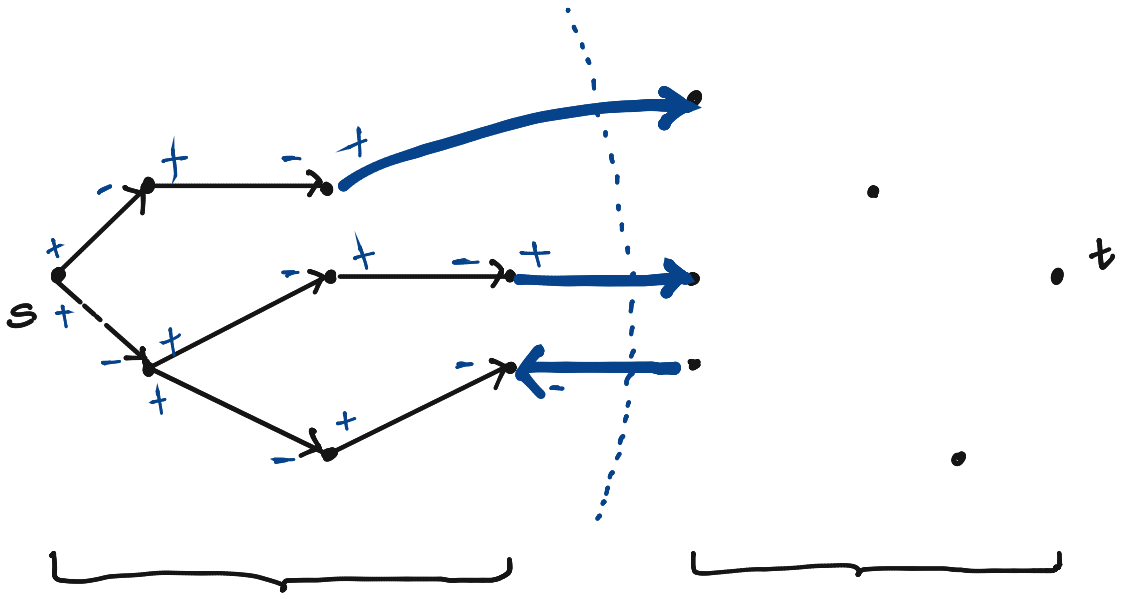
$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e) = \text{FLOW FROM } s \text{ to } t$$

FOR ANY OTHER INTERNAL NODE v IN A,

$$\sum_{e \text{ OUT OF } v} f(e) - \sum_{e \text{ IN } v} f(e) = 0$$

LEMMA: LET f BE A st FLOW AND $c(A,B)$ BE st CUT, THEN

$$\text{FLOW FROM } s \text{ to } t = \sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e)$$



$$\sum_{e \text{ OUT OF } s} f(e) - \sum_{e \text{ IN } s} f(e) = \text{FLOW FROM } s \text{ to } t$$

FOR ANY OTHER INTERNAL NODE v IN A,

$$\sum_{e \text{ OUT OF } v} f(e) - \sum_{e \text{ IN } v} f(e) = 0$$

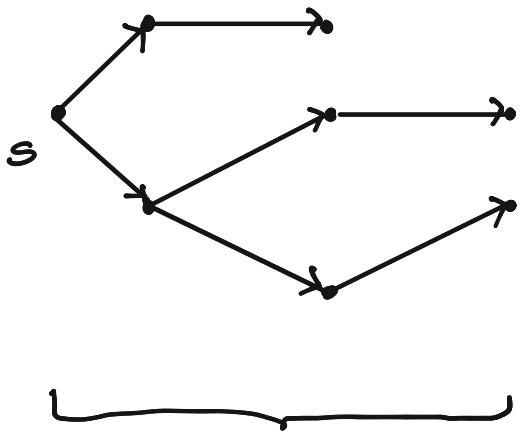
$$\sum_{e \text{ OUT of } A} f(e) - \sum_{e \text{ IN } A} f(e) = \text{FLOW FROM } s \text{ to } t$$

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

→ NOW LET US LOOK AT OUR ALGO.

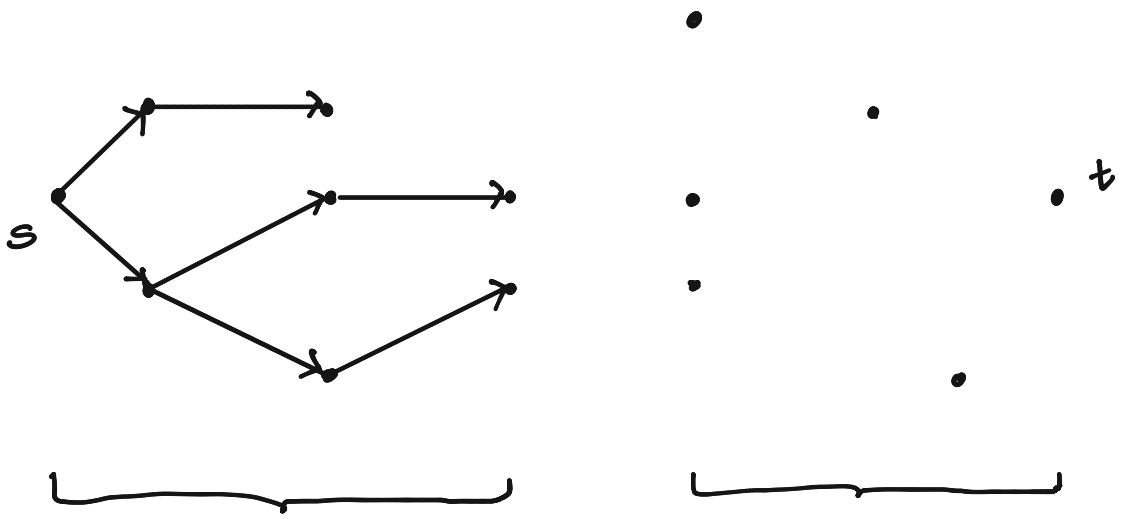
→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



VERTICES CONNECTED TO s IN G_R

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

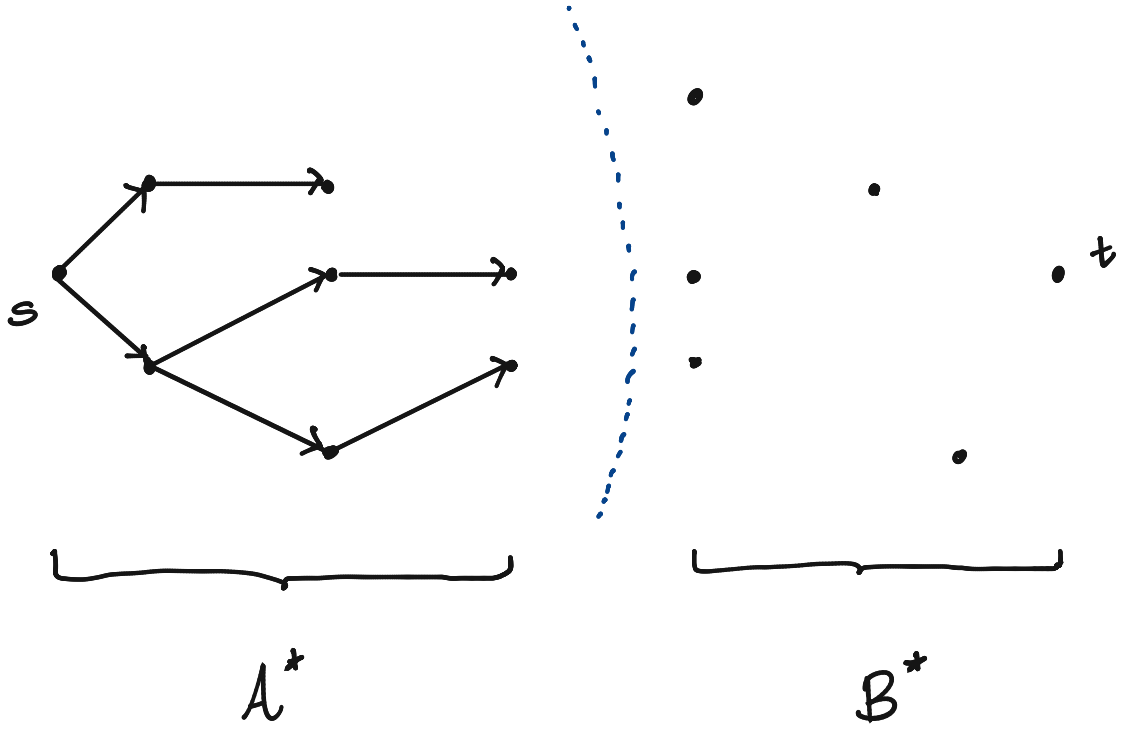


VERTICES CONNECTED TO s IN G_R

VERTICES NOT CONNECTED TO s IN G_R

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

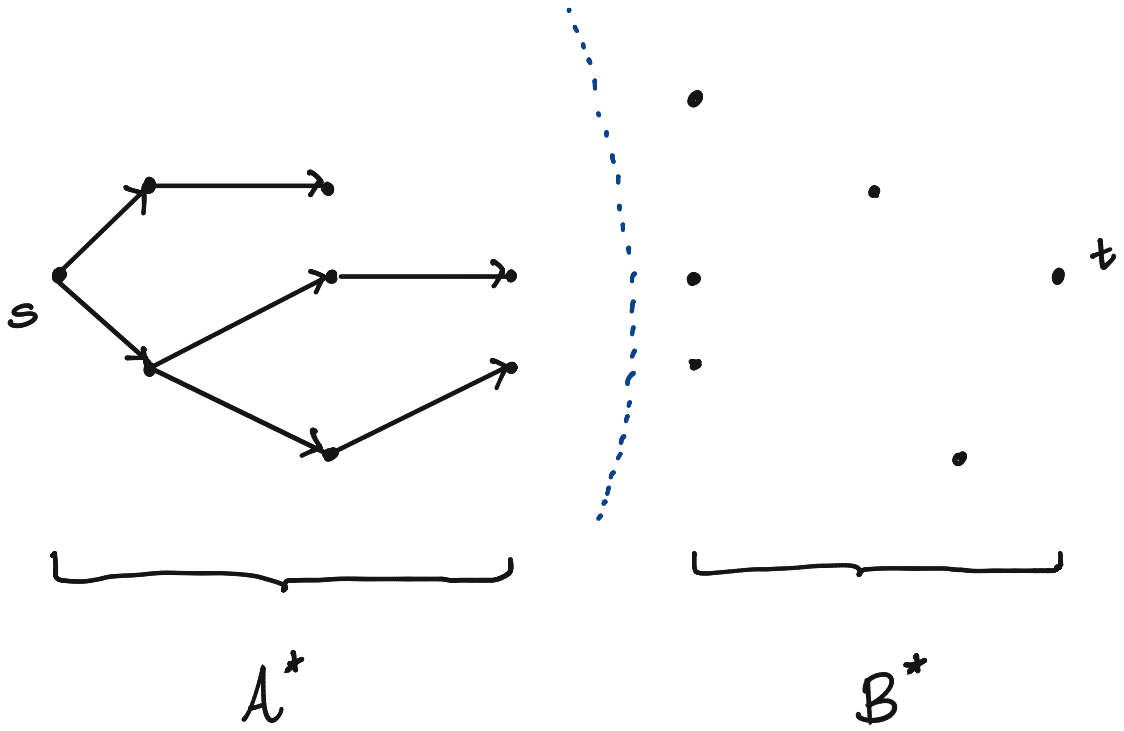


BY OUR LEMMA, FLOW FROM s to t

$\leq$

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

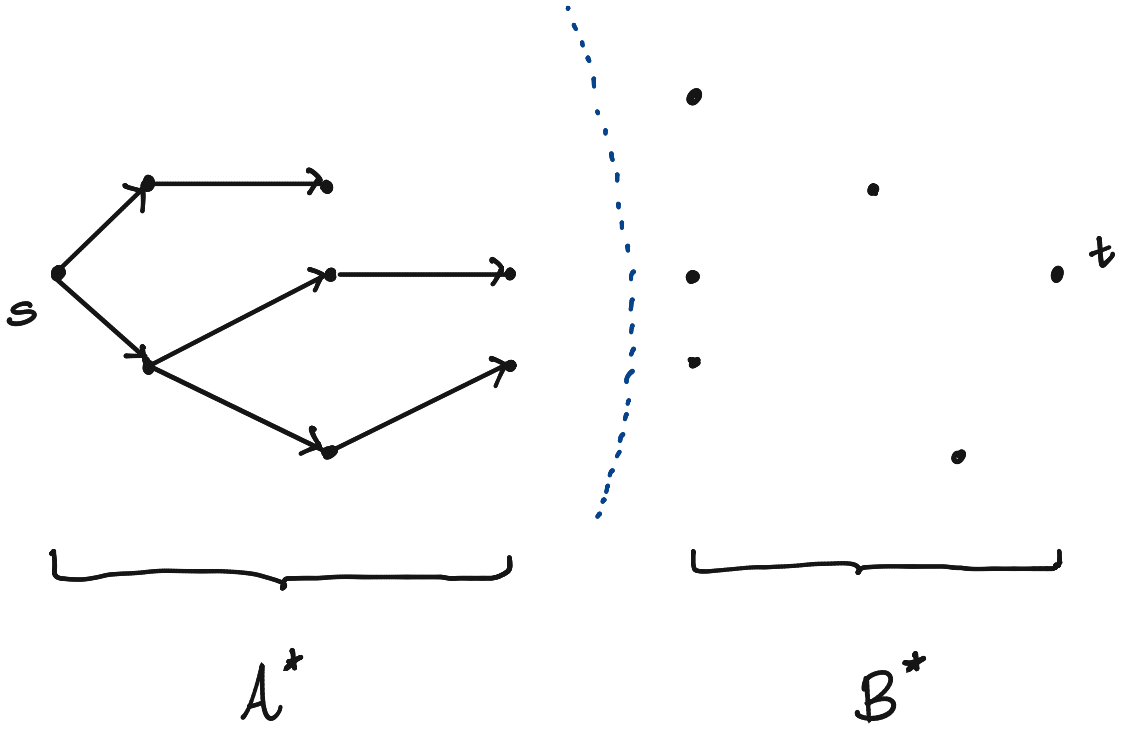


BY OUR LEMMA, FLOW FROM s to t

$\leq$ WEIGHT OF CUT $c(A^*, B^*)$.

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



BY OUR LEMMA, FLOW FROM s to t $\leq$ WEIGHT OF CUT $c(A^*, B^*)$.

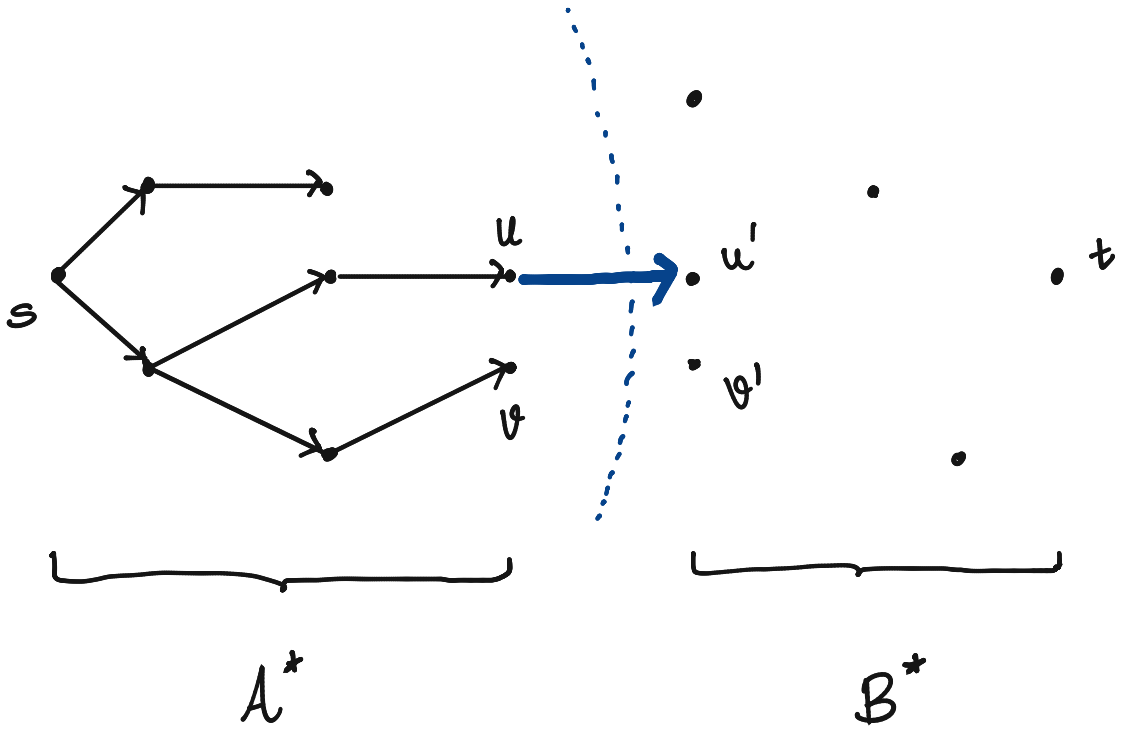
WE WILL PROVE

FLOW FROM s to t = WT. OF MINIMUM CUT

$c(A^*, B^*)$ IS THE MINIMUM st CUT.

→ NOW LET US LOOK AT OUR ALGO.

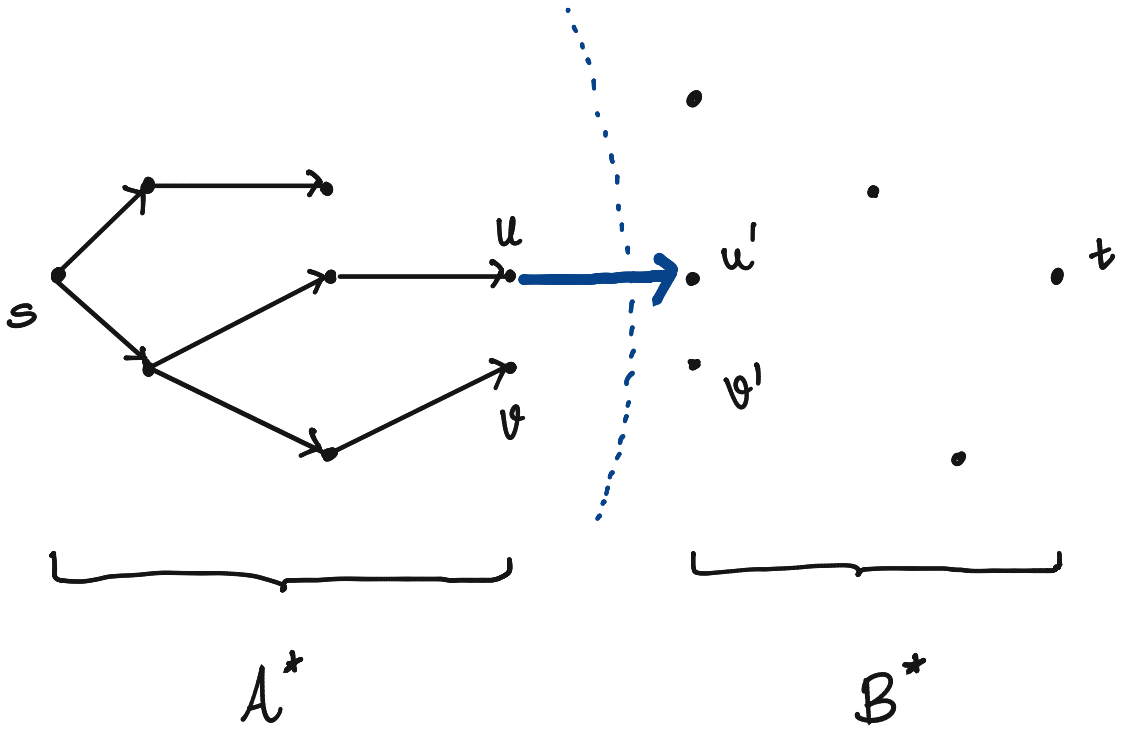
→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



Q WHAT IS THE FLOW THROUGH (u,u') AT THE END OF OUR ALGO ?

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



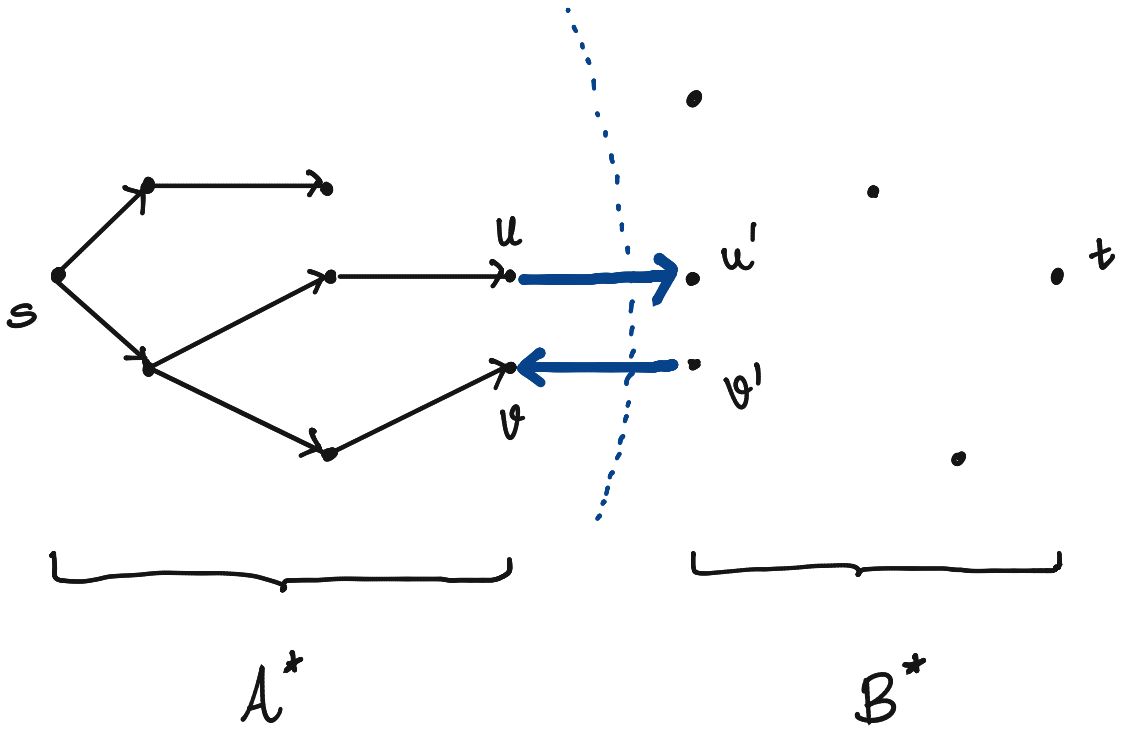
Q WHAT IS THE FLOW THROUGH (u,u') AT THE END OF OUR ALGO ?

A (u,u') IS SATURATED

$f(u,u') = c(u,u')$

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

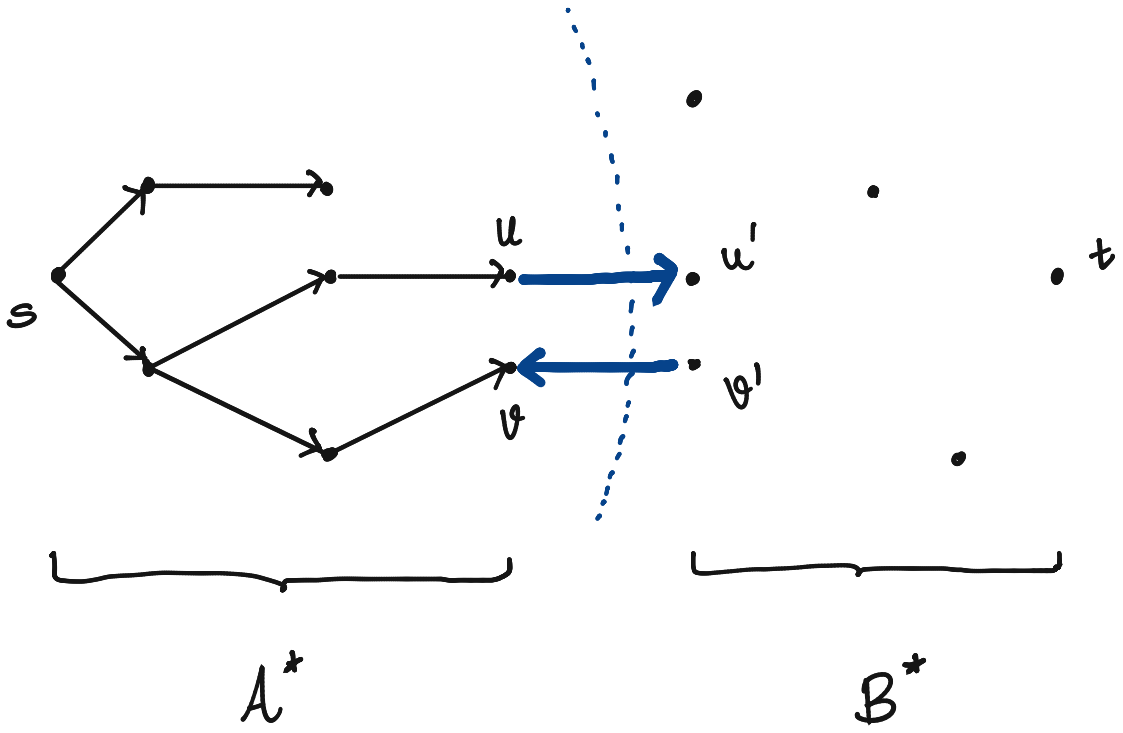


Q WHAT IS THE FLOW THROUGH (v, v') AT THE END OF OUR ALGO ?

A

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.

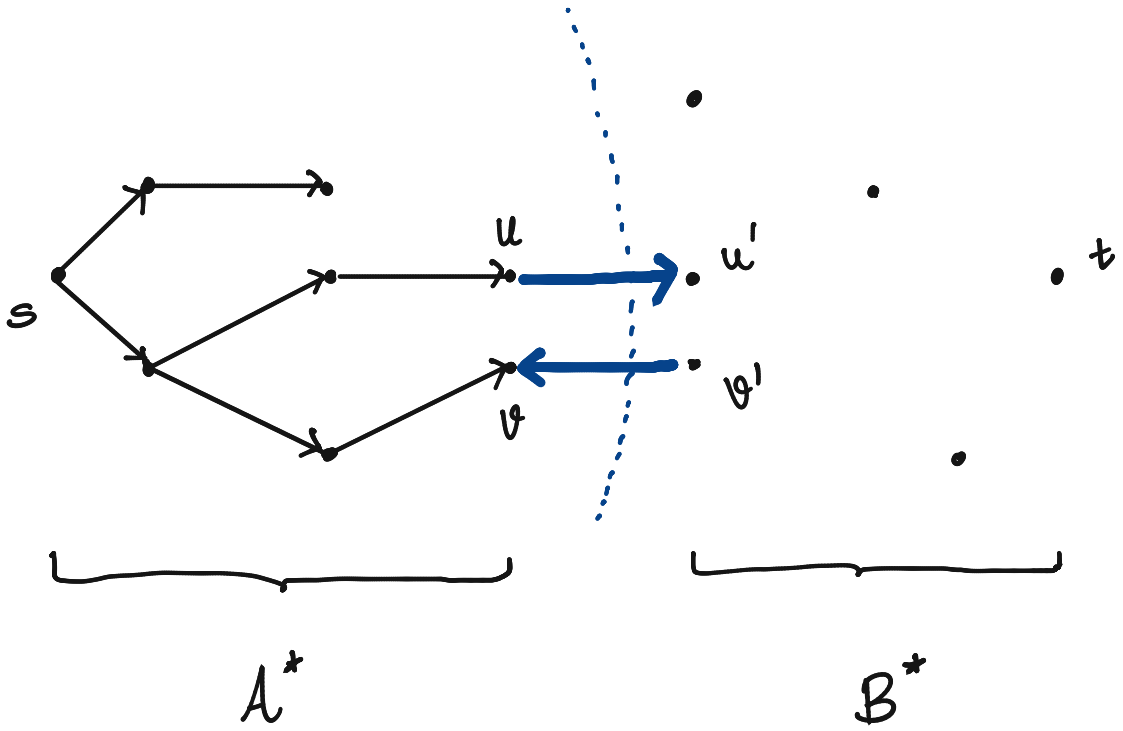


Q WHAT IS THE FLOW THROUGH (v, v') AT THE END OF OUR ALGO ?

A · $f(v, v') = 0$

→ NOW LET US LOOK AT OUR ALGO.

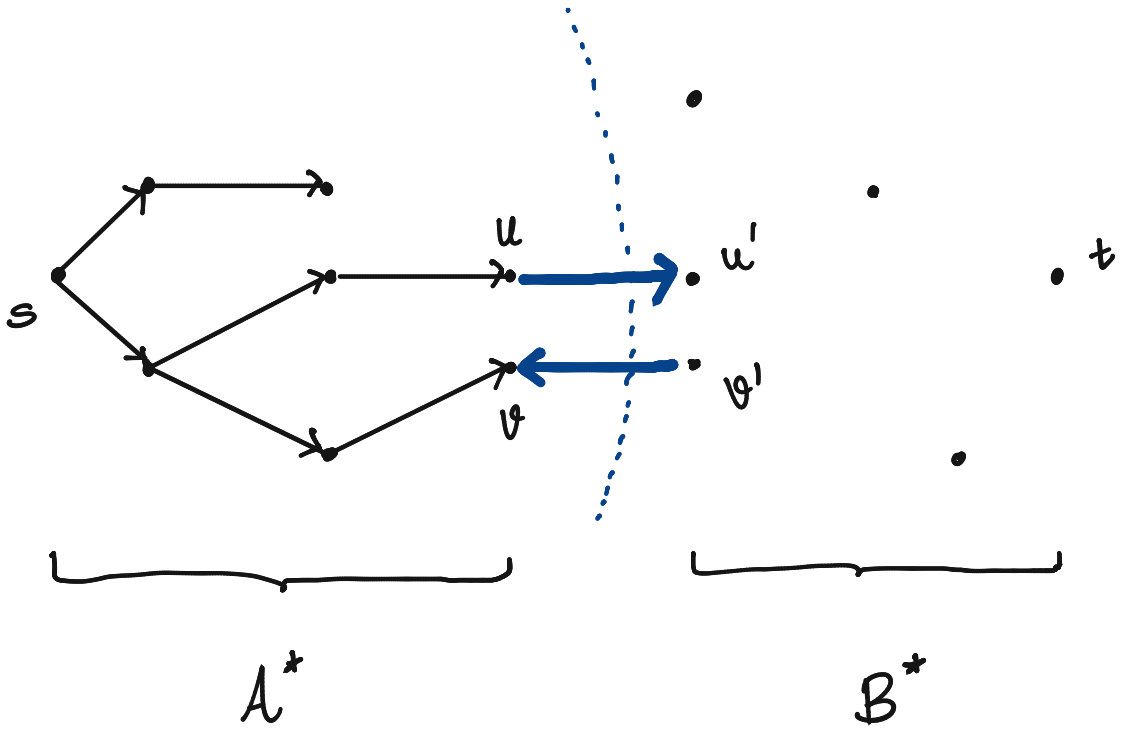
→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



FLOW FROM s to t $= \sum_{e \text{ OUT OF } A^*} f(e) - \sum_{e \text{ IN } A^*} f(e)$

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



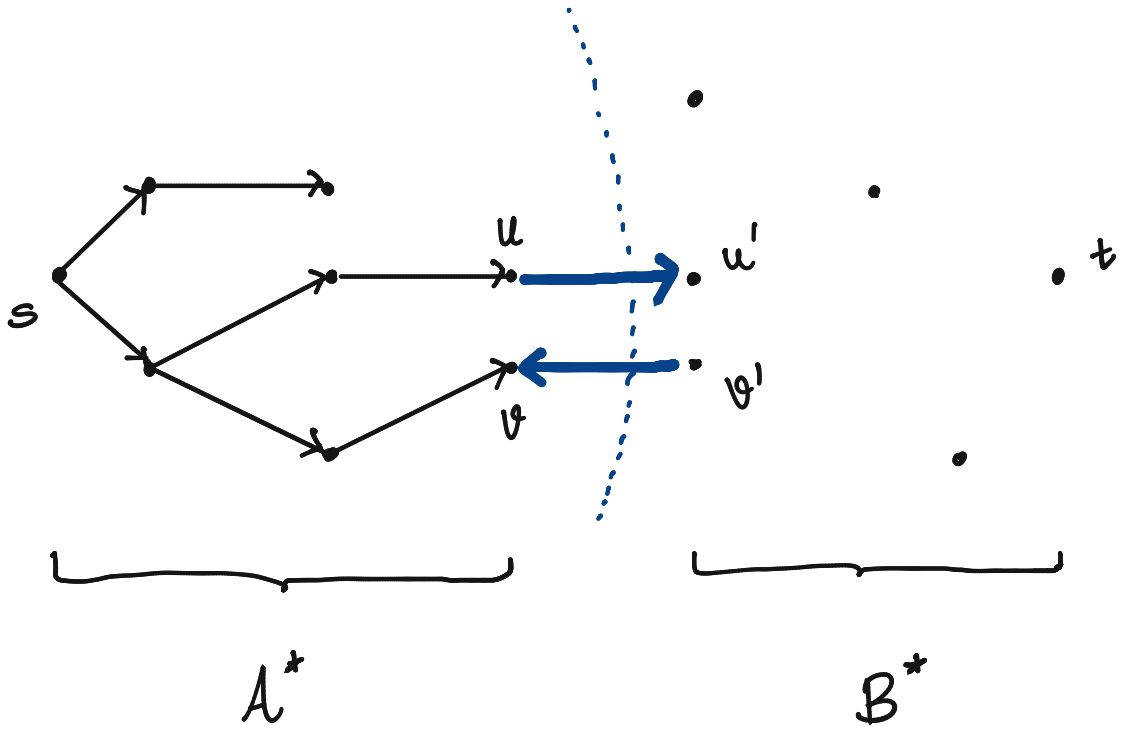
FLOW FROM s to t $= \sum_{e \text{ OUT OF } A^*} f(e) - \sum_{e \text{ IN } A^*} f(e)$

$= \sum_{e \text{ out of } A^*} c(e)$

= WEIGHT OF $c(A^*, B^*)$

→ NOW LET US LOOK AT OUR ALGO.

→ AFTER SOME ITERATION, WE ARE LEFT WITH A RESIDUAL GRAPH SUCH THAT THERE IS NO st PATH IN G_R.



FLOW FROM s to t = WEIGHT OF CUT $c(A^*, B^*)$

BUT FLOW FROM s to t $\leq$ WEIGHT OF ANY CUT $c(A, B)$

$\Rightarrow c(A^*, B^*)$ IS THE MINIMUM WEIGHT s-t CUT.

→ ANY s-t FLOW ≤ WEIGHT OF MINIMUM st CUT

→ OUR ALGORITHM RETURNS A FLOW = WEIGHT OF MINIMUM st CUT

⇒ OUR ALGORITHM RETURNS THE MAXIMUM FLOW

→ ANY s-t FLOW ≤ WEIGHT OF MINIMUM st CUT

→ OUR ALGORITHM RETURNS A FLOW = WEIGHT OF MINIMUM st CUT

⇒ OUR ALGORITHM RETURNS THE MAXIMUM FLOW

FORD - FULKERSON ALGORITHM.